VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF BIOMEDICAL ENGINEERING

SYSTÉM PRO MĚŘENÍ AEROBNÍ STABILITY FERMENTOVANÝCH KRMIV

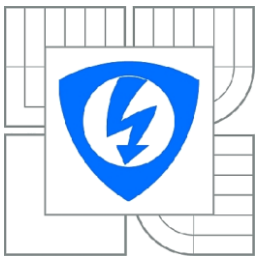
SYSTEM FOR MEASUREMENT OF AEROBIC STABILITY OF FORAGE

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

AUTOR PRÁCE — Bc. JOSEF SYNEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. VRATISLAV HARABIŠ
SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav biomedicínského inženýrství

Diplomová práce

magisterský navazující studijní obor
Biomedicínské inženýrství a bioinformatika

Student:	Bc. Josef Synek	***ID:***	109614
Ročník:	2	***Akademický rok:***	2011/2012

NÁZEV TÉMATU:

Systém pro měření aerobní stability fermentovaných krmiv

POKYNY PRO VYPRACOVÁNÍ:

1) Seznamte se s teorií a vypracujte literární rešerši výroby fermentovaných krmiv, zaměřte se na sledování významných veličin při výrobě, zejména na aerobní stabilitu. 2) Navrhněte systém pro měření a analýzu aerobní stability fermentovaných krmiv. 3) Navrhněte hardwarovou část umožňující snímat teplotu pomocí několika teplotních čidel DS18S20, v souladu s požadavky firmy NutriVet,s.r.o. 4) Ve zvoleném programovacím prostředí vytvořte software, který umožní jak ovládání hardwarové části, tak i zpracování a vizualizaci naměřených dat. Software by měl umožnit zobrazení průběhu teplot z jednotlivých čidel v čase a výpočet průměrných teplot. 5) Následně software doplňte o část umožňující výpočet doby stability fermentovaných krmiv. 6) Proveďte diskuzi navrženého softwarového a hardwarového řešení. Funkci navrženého systému otestujte a proveďte zhodnocení dosažených výsledků.

DOPORUČENÁ LITERATURA:

[1] KUNG, L. Aerobic Stability of Silage, The 2010 California Alfalfa and Forage Symposium, Visalia, California.
[2] MAXIM DS18S20 High-Precision 1-Wire Digital Thermometer datasheet [online], Rev 8/10 [cit. 25. Srpna 2011]. Dostupné z <http://datasheets.maxim-ic.com/en/ds/DS18S20.pdf>.

Termín zadání:	6.2.2012	***Termín odevzdání:***	18.5.2012

Vedoucí práce: Ing. Vratislav Harabiš
Konzultanti diplomové práce:

prof. Ing. Ivo Provazník, Ph.D.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

Abstrakt

Aerobní stabilita je termín, který odborníci na výživu používají k určení doby, po kterou krmivo zůstává stabilní tedy nezkažené. Jakmile je fermentace dokončena a krmiva jsou vystavena vzduchu během krmení nebo při skladování (netěsnost sil, špatné balení) dochází ke zvýšení teploty těchto krmiv působením kvasinek nebo v menší míře působením některých bakterií. Tuto teplotu měříme pomocí teplotních čidel a porovnáváme ji s ambientní teplotou okolí. Aerobní stabilita se udává jako čas, za který teplota krmiva vzroste o 3°C nad ambientní teplotu.

Klíčová slova: aerobní stabilita, krmiva, teplota, teplotní čidla

Abstract

Aerobic stability is a term which nutritionists use to define the length of time that forage remains stabil and don‘t degrade. Once fermentation is completed and the forage is exposed to air during feeding or during storage (leaky silos, poor packaging) increases the temperature of the forage initiated by yeasts or lesser extent by the action of some bacterias. This temperature is measured using temperature sensors and it is compared with the ambient temperature of the surroundings. Aerobic stability is given as the time, for which the forage temperature rises by 3°C above ambient temperature.

Keywords: aerobic stability, forage, temperature, temperature probe

Bibliografická citace projektu:

SYNEK, J. *Systém pro měření aerobní stability fermentovaných krmiv.* Brno: FEKT VUT v Brně, 2012. 73 s., 3 příl.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Systém pro měření fermentovaných krmiv jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ... .

podpis autora

Poděkování

Děkuji Ing.Vratislavu Harabišovi a Ing.Václavu Jamborovi, CSc. za jejich pomoc a cenné rady při tvorbě a zpracování mé diplomové práce. Dále bych chtěl poděkovat své rodině, přátelům a všem co mi pomáhali během mého studia.

V Brně dne ... .

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

DM ……sušina
pH ……logaritmus aktivity oxoniových kationtů
ADP ……...adenosin difosfát
P……...fosfát
CFU……..kolonie tvořící jednotka (colony forming units)

1. Úvod

Aerobní stabilita je vlastnost krmiva udržet za přístupu vzduchu svoji kvalitu co nejdelší dobu s co možná nejmenšími ztrátami.

Kvalitní krmivo je nezbytnou součástí každého moderního zemědělského podniku. Spolu s mírou kvality krmiv se mění také ekonomické výsledky. Mezi používaná potom patří zejména fermentovaná krmiva. Původním termínem pro všechna fermentovaná krmiva je siláž. Silážování je způsob konzervování krmiva ve šťavnatém stavu. Siláž zachovává jak obsah živin, tak i vitamínů a minerálních látek použitého materiálu. Konzervace probíhá působením mléčného kvašení cukrů obsažených v píci. Celý proces musí probíhat bez přístupu vzduchu. [1]

Silážování je technologicky náročný proces a při nedodržení správných podmínek nebo použití nevhodných surovin a konzervačních přípravků dochází k aerobní degradaci. Potom je znehodnocena biochemickými pochody, kdy vzniká kyselina máselná, octová nebo mravenčí a siláž nepříjemně páchne a je dále nepoužitelná.

O řešení aerobní stability siláží, především těch kukuřičných, usiluje ve světě mnoho vědců i farmářů. Je to velký problém. Zemědělci vyrobí kvalitní siláž a pak, krátce před tím, než se zkrmí dobytku, se z ní vytratí energie ve formě tepla, které vzniká především činností kvasinek.

Aerobní stabilitu siláží lze sledovat různými způsoby. Z pohledu výsledků a důkazů původu aerobní nestability je nejlepší stanovení změny počtu mikroorganismů, které způsobují sekundární fermentaci. Toto stanovení je však velmi drahé a v odborných kruzích je princip sekundární fermentace již dokázán. My se zde zaměříme na sledování ukazatele, který je doprovodným jevem zvýšené mikrobiální aktivity. Tímto ukazatelem je teplota siláže, která je lehce měřitelná. Je to tedy ukazatel, který rozhoduje o tom, zda siláž je stabilní (teplota je konstantní), nebo zda se teplota zvyšuje. Proto se nám jeví tento ukazatel jako velmi vhodný pro sledování sekundární fermentace. [2]

Cílem práce je zhotovit systém pro měření aerobní stability fermentovaných krmiv. Tento systém se bude sestávat z několika teplotních čidel, která budou zaznamenávat teplotu v čase a porovnávat ji s teplotou okolí. Výsledkem potom bude čas, za který se vzorky ohřejí o tři stupně celsia vlivem sekundární fermentace oproti jejich počáteční teplotě.

2. Fermentovaná krmiva

Objemná krmiva představují více než polovinu krmné dávky ve výživě skotu. Aby bylo zajištěno kvalitní krmení s vysokou výživnou hodnotou, je nutné tato objemná krmiva konzervovat. V současnosti se objemná krmiva konzervují sušením (seno, sláma) nebo silážováním. Metodou silážování se konzervuje více než 75 % objemných krmiv. [3] Je to technologie založená na rychlém okyselení naskladněné, pořezané a dobře udusané hmoty ve šťavnatém nebo zavadlém stavu za anaerobních podmínek. Kvalita siláží je ovlivněna jak obsahem a poměrem živin konzervované píce, tak především i vlastním průběhem fermentačního procesu a podmínkami skladování. V průběhu fermentace dochází k tvorbě organických kyselin, zejména pak kyseliny mléčné z jednoduchých cukrů a tím k okyselování silážované hmoty a její konzervaci. Siláž, výsledek procesu silážování, pak můžeme popsat jako šťavnaté, kyselé, nebo mírně nakyslé krmivo, které by se mělo vyznačovat příjemnou vůní po původní hmotě. [3]

Siláže slouží především jako náhrada pastvy v zimních měsících. Může být ale podávána po dobu celého roku. [4]

2.1 Silážování – základní pojmy

V samotném úvodu tohoto textu musíme zavést některé pojmy, bez kterých bychom se dále neobešli.

2.1.1 Sušina

Z anglického slova „dry matter“ vznikla zkratka DM. Český výraz pro DM je sušina. Jde vlastně o procentuální podíl suché části v určitém objemu. Pokud je hodnota DM 40 %, pak krmivo obsahuje 40 % sušiny a 60 % vody. Pokud hodnota sušiny přesáhne 50%, pak nejčastěji hovoříme o senáži. [1]

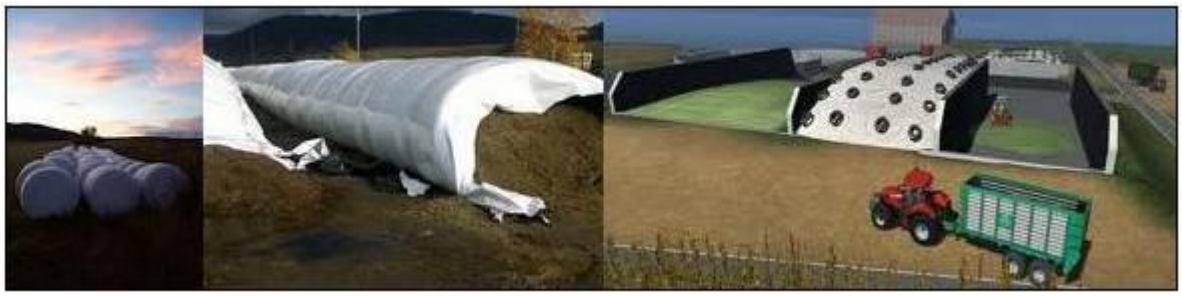
2.1.2 pH

Jde o záporný dekadický logaritmus koncentrace vodíkových kationtů, hodnota udávající, zda roztok reaguje kysele nebo zásaditě. Je to faktor ukazující úspěšnost fermentace. Siláž by měla mít pH okolo 4,0. Takováto siláž je potom mnohem lépe konzervačně chráněna než jiná s hodnotou pH např. 6,5, protože organismům zapříčiňujícím kažení se ve více kyselých prostředí nedaří.

2.1.3 Způsoby silážování

Siláž se dá uskladňovat různými způsoby. V našich podmínkách se nejčastěji využívají takzvané silážní žlaby nebo jámy. Mají různou kapacitu a konstrukčně jsou velmi pestré, jsou buď průjezdné, nebo neprůjezdné, nadzemní, polozapuštěné, zapuštěné. Další hojně užívaný postup je silážování do vaků, přičemž je silážovaná hmota natlačena do rukávců o délce až 60 m s průměrem až 2,5 metrů a kapacitě do 200 tun. Tyto igelitové rukávce se vyznačují

většinou absolutní nepropustností světla a vzduchu při malých investičních nákladech. Jinou možností je silážní věž, zaručující kvalitní výsledky avšak za velkou investici. Tato metoda se hodí pro automatizované systémy krmení. Dále existuje možnost silážovat do balíků potažených folií. [7]



Obrázek 1: Druhy uskladnění siláží [7][12][13]

2.2 Plodiny vhodné pro silážování

Silážováním lze úspěšně konzervovat jak jednoleté i dvouleté pícniny, tak i některá krmiva potravinářského průmyslu, pokud dodržíme správný technologický postup.

Z víceletých pícnin se nejčastěji silážují:

- **Jeteloviny** (vojtěška, jetel luční), tato krmiva mají bílkovinnou povahu a mají nízký obsah vodorozpustných sacharidů, proto se před silážováním nechávají zavadat na vyšší sušinu (35 – 50 %).
- **Jetelotrávy** jsou potom převážně polobílkovinná krmiva, optimální sušina je 35–40 %.
- **Trávy**, které mají povahu glycidového až polobílkovinného krmiva, mají vyšší obsah lehce fermentovatelných cukrů a zavadají se na sušinu 30-35 %.

Z jednoletých pícnin je nejznámější siláž ze silážní kukuřice, ale silážují se i drtě celých rostlin luskovin (hrách, bob) a obilovin (pšenice, ječmen, oves).

Mezi krmiva potravinářského průmyslu, která lze silážovat patří cukrovarské řízky nebo pivovarské mláto. Silážovat je možné i další suroviny, které mají dostatečný obsah vodorozpustných sacharidů nezbytných pro fermentaci a vznik kvasných kyselin. [3]

2.3 Faktory ovlivňující kvalitu silážování

Na kvalitu výsledné siláže má vliv řada faktorů ovlivnitelných různou měrou. Hlavní faktory přímo ovlivňující výživnou hodnotu siláže při sklizni plodiny jsou:

- **Vegetační stádium porostu** – množství vlákniny, škrobu, stravitelnost organických živin a další.

- **Termín sečení porostu v průběhu dne**:
    - Ráno – málo vodorozpustných cukrů v rostlině, rosa – snižuje obsah sušiny.
    - Odpoledne – při slunečném počasí je intenzivní fotosyntéza – vyšší obsah cukrů a následně intenzivní zavadání.
    - Večer – v rostlině je poměrně velké množství cukrů, ale v noci je zavadání málo intenzivní a dochází k jejich ztrátě.
- **Výška strniště** – případná kontaminace půdou, intenzita zavadání
- **Počasí** – velice důležitý faktor při sklizni a silážování, který nelze ovlivnit, ale lze mu přizpůsobit technologický postup.

Kvalitu siláže, respektive fermentačního procesu ovlivňují i jiné faktory jako je sušina silážované hmoty, délka řezanky, intenzita dusání, kvalita a rychlost zakrytí a utěsnění sila.[3]

2.4 Podmínky dobrého silážování

Silážování je konzervace zelené píce pomocí bakterií mléčného kvašení za anaerobních podmínek. Aby byla konzervace úspěšná, je třeba splnit následující podmínky:

- Přítomnost dostatečně velkého množství zkvasitelných cukrů, aby konečné pH kleslo na 4,0 – 4,2. Toto je ovlivněno hlavně použitým druhem pícniny. Nejlepší je kukuřice v takzvaně mléčně-voskové zralosti. Pokud je surovina na tyto fermentativní cukry chudá, je nutné je dodat například ve formě melasy.
- Přítomnost bakterií mléčného kvašení. Ty za anaerobních podmínek pomocí homo a heterofermentativního mléčného kvašení vytvoří kyselinu mléčnou prostupující a konzervující rostlinou hmotu.
- Anaerobní podmínky musí být zajištěny tak, že rostlinná hmota je nařezána na drobné kousky a je důkladně utlačena do žlabů nebo vaků utěsněných folií. [4]

3. Působení organismů v průběhu silážování

Mikroorganismy mají v průběhu konzervačního procesu nezbytnou roli. Mikroflóra siláží se obvykle dělí na dvě skupiny. První jsou prospěšné a žádoucí a patří sem především bakterie mléčného kvašení. Druhou skupinu tvoří bakterie způsobující kažení za anaerobních podmínek, jako jsou klostridie a enterobakterie. Za anaerobních podmínek působí také kvasinky a plísně. Tyto nežádoucí organismy snižují obsah živin a chutnost siláží, a tak snižují kvalitu nebo mohou představovat riziko pro zvířata a potažmo i člověka. Dále mají negativní vliv na kvalitu mléka a mléčných výrobků. [4]

3.1 Přirozená mikroflóra siláží

Mikroflóra siláží závisí především na složení tzv. epifytní mikroflóry na povrchu rostlin. Její složení je uvedeno v následující tabulce.

Tabulka 1: Složení epifytní mikroflóry [4]

Skupina	Počet v logCFU/g
Aerobní bakterie	> 7
Bakterie mléčného kvašení	1 -6
Koliformní bakterie	3 - 6
Kvasinky	3 - 5
Plísně	3 - 4
Klostridie (spory)	2 - 3
Bacily (spory)	2 - 3

Jak vyplývá z údajů, počty bakterií mléčného kvašení jsou nejvíce variabilní. Počty koliformních bakterií kolísají také výrazně v závislosti na intenzitě a způsobu hnojení plodin. Jak bylo již zmíněno, přirozená mikroflóra siláže obsahuje jak mikroflóru žádoucí, tak i tu nežádoucí. Společným znakem těchto mikroorganismů je to, že jsou zpravidla vždy, i když v různé míře, přítomny a musíme tedy počítat s jejich pozitivní i negativní aktivitou. Tabulka 2 zobrazuje hlavní skupiny přirozené mikroflóry.

Tabulka 2: Hlavní skupiny mikroorganismů účastnící se fermentačních pochodů v siláži

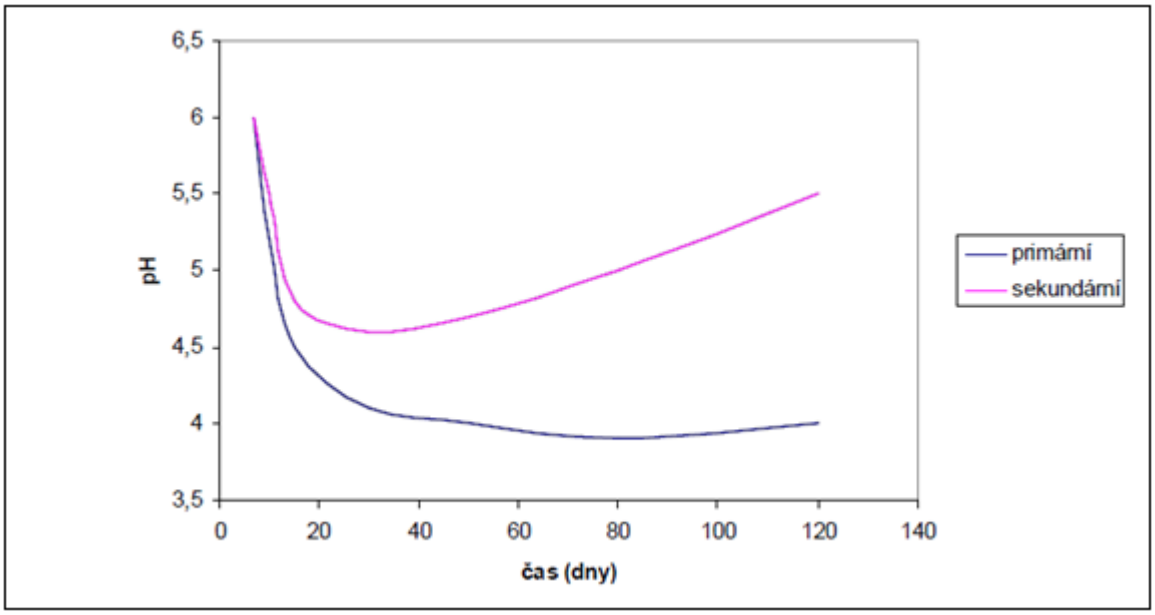
Druh	Zdroj	Substrát	Metabolity
Enterobakterie (koliformní bakterie)	Splašky, chlévská mrva, půda	Vodorozpustné cukry	Kyselina octová, etanol, CO_2, amoniak
Kvasinky	Povrch rostlin, obiloviny	Vodorozpustné cukry	Etanol, CO_2
Homofermentativní	Povrch rostlin,	Vodorozpustné cukry	Kyselina mléčná

BMK	obiloviny		
Heterofermentativní BMK	Povrch rostlin, obiloviny	Vodorozpustné cukry	Kyselina mléčná, kyselina octová, etanol, manitou, CO_2
Klostridie	Půda	Kyselina mléčná, bílkoviny, aminokyseliny	Kyselina máselná, kyselina octová, CO_2, H_2, aceton

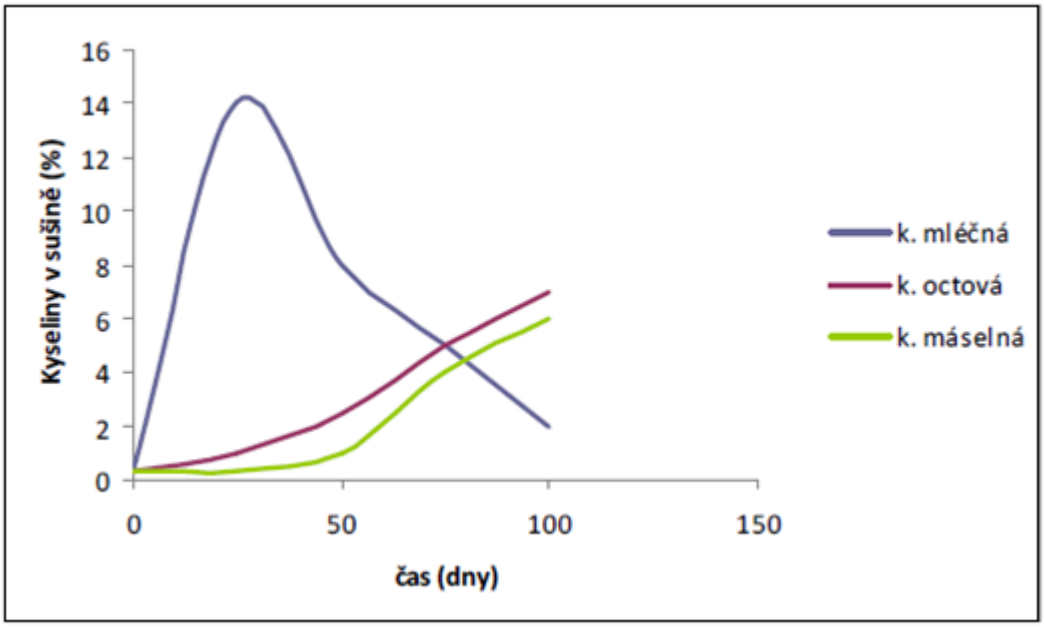
3.2 Pochody při silážování

Pokud proběhne celý proces silážování optimálním způsobem, uskuteční se pouze tzv. primární kvašení. Potom hodnota pH klesne na hodnotu 4,0 – 4,2 a vytvoří se cca 1,7 % kyseliny mléčné, 0,7 % kyseliny octové a hodnota kyseliny máselné je do 0,3 %. Siláž s takovými vlastnostmi za optimálních podmínek skladování vydrží dlouhodobě stabilní bez větších chemických změn nejméně 3 – 4 měsíce.

Pokud z různých příčin (nedostatečná mikroflóra, obsah pufrujících látek, ale hlavně nedostatek zkvasitelných cukrů) neproběhne primární fermentace důkladně, zpravidla následuje sekundární kvašení. Toho se účastní hlavně klostridie a někdy také koliformní bakterie. Při sekundárním kvašení dochází ke zvyšování pH následkem fermentace dvou molekul relativně silné kyseliny mléčné nebo octové oproti jedné molekule slabší kyseliny máselné a také proto, že kvašením aminokyselin a rozkladem bílkovin vzniká amoniak. Následující grafy zobrazují změny pH v průběhu primárního a sekundárního kvašení a změny koncentrací jednotlivých organických kyselin v průběhu sekundární fermentace v závislosti na čase. [4]



Obrázek 2: Změny pH siláže v průběhu primárního a sekundárního kvašení [4]



Obrázek 3: Změny koncentrací organických kyselin v průběhu sekundárního kvašení [4]

V průběhu silážování se také mění složení dusíkatých látek a to působením klostridií, laktobacilů a v menší míře i ostatních bakterií mléčného kvašení. Obsah z celkového dusíku klesá ze zhruba 90 % v surovině na méně než 60 % v siláži. Obsah aminokyselin naopak stoupá z cca 10 % na téměř 40 %. Následkem silážování se také zvyšuje obsah amoniaku z 0 na cca 5%. K větším biochemickým změnám potom dochází při vystavení siláže působení vzduchu. Nejvíce aktivní jsou enterobakterie (koliformní bakterie) a bakterie mléčného kvašení. Také může dojít k pomnožení kvasinek a plísní.

3.2.1 Koliformní bakterie

Koliformní bakterie, nebo enterobakterie jsou zastoupeny *Escherichia coli* a příbuznými rody jako je *Enterobacter, Erwinia* a další. Hlavní metabolickou činností v siláži je konverse glukosy na acetát a etanol podle rovnice:

$$\text{Glukosa} + 3\text{ADP} + \text{P} \rightarrow \text{acetát} + \text{etanol} + 2\text{CO}_2 + 2\ \text{H}_2 + 3\ \text{ATP} + 2\text{H}_2\text{O} \qquad (1)$$

Jak vyplývá z rovnice v důsledku fermentace dochází ke ztrátám uhlíku (ve formě plynu) a protože z jednoho molu glukosy vzniká pouze jeden mol acetátu, je proces okyselování nedostatečný. Koliformní bakterie se nemnoží při pH<5, z toho důvodu je důležité rychlé okyselení při silážování.

3.2.2 Bakterie mléčného kvašení

Bakterie mléčného kvašení, které se uplatňují při silážování, patří hlavně mezi epifytní mikroflóru, z čehož plyne, že se vyskytují na povrchu rostlin. Při silážování se uplatňují tři skupiny bakterií mléčného kvašení:

- **Obligátně homofermentativní**

Mezi obligátně homofermentativní bakterie mléčného kvašení patří např. *Lactobacilus acidophilus, Pediococcus damnosus* a *Lactococcus lactis*. Tyto bakterie jsou žádoucí mikroflórou siláže. Bývají také součástí silážních inokulantů. Jeden mol glukosy nebo fruktosy fermentují vždy na dva moly kyseliny mléčné podle rovnice:

$$\text{Glukosa nebo fruktosa} + 2\text{ADP} + \text{P} \rightarrow 2\ \text{laktát} + 2\text{ATP} + 2\text{H}_2\text{O} \qquad (2)$$

Jak vyplývá z rovnice, nedochází ke ztrátám uhlíku, pokud nepočítáme možný únik silážních šťáv. Určitou nevýhodu těchto bakterií je, že většinou nejsou schopny využívat pentosy, např. xylosu.

- **Fakultativně heterofermentativní**

Jako jsou *Lactobacillus plantarum* a *L. casei* a patří mezi nejvíce žádoucí a nejdůležitější bakterie siláží, proto jsou také využívány jako silážní inokulanty. Fakultativně heterofermentativní bakterie fermentují hexosy (glukosa, fructosa) stejně jako homofermentativní mléčné bakterie, ale pentosy (xylosa, arabinosa) na laktát, acetát a někdy i etanol.

- **Obligátně heterofermentativní**

Jsou poslední skupinou mléčných bakterií a jsou zastoupeny *L. buchneri, L. brevis* a *Leuconostoc mesenteroides*. Tyto bakterie tvoří kromě kyseliny mléčné i další metabolity podle rovnic:

$$\text{Glukosa} + \text{ADP} + \text{P} \rightarrow \text{laktát} + \text{etanol} + \text{CO}_2 + \text{ATP} + \text{H}_2\text{O} \qquad (3)$$

$$3\ \text{Fructosa} + 2\ \text{ADP} + 2\text{P} \rightarrow \text{laktát} + \text{acetát} + 2\ \text{manitol} + \text{CO}_2 + 2\ \text{ATP} + \text{H}_2\text{O} \qquad (4)$$

Heterofermentativní mléčné bakterie jsou v silážích méně žádoucí, protože produkují méně kyselin. V poslední době je ale zdůrazňován pozitivní vliv kyseliny octové na aerobní stabilitu siláže, proto je např. L. buchneri také používán jako silážní inokulant. [4]

3.2.3 Kvasinky

Jde o eukaryotní, fakultativně anaerobní mikroorganismy. V silážích se vyskytuje několik rodů, především *Kandida, Hansenula, Saccharomyces* a *Torulopsis*. Za anaerobních podmínek provádějí alkoholové kvašení podle rovnice:

$$\text{Glukosa} + 2\ \text{ADP} + 2\ \text{P} \rightarrow 2\ \text{etanol} + 2\ CO_2 + 2\ \text{ATP} \qquad (5)$$

Alkoholové kvašení není pro siláž vhodné, protože nevzniká žádná kyselina, dochází také ke ztrátám uhlíku a navíc některé kvasinky mohou sekundárně fermentovat i kyselinu mléčnou. Za aerobních podmínek kvasinky oxidují kyselinu mléčnou na vodu a oxid uhličitý. Proto se zvyšuje pH a vzniká půda pro kažení dalšími mikroorganismy.

3.2.4 Klostridie

Klostridie jsou v siláži považovány za nežádoucí, ale jsou prakticky vždy přítomné. Účastní se v různé míře anaerobních pochodů a proto je nutno považovat za přirozenou mikroflóru. Hlavními druhy jsou *Clostridium tyrobutyricium, C. butyricum* a *C. bifermentas*. V siláži se klostridie uplatňují zejména při sekundárním kvašení, když nedojde k rychlé tvorbě kyselin a pH je větší než 4,6, kde jako substrát používají kyselinu mléčnou podle následující rovnice:

$$2\ \text{Laktát} + \text{ADP} + \text{P} \rightarrow\ \text{Butyrát} + 2\ CO_2 + 2\ H_2 + \text{ATP} + H_2O \qquad (6)$$

Tato činnost je z hlediska kvality siláže vesměs negativní, protože dochází ke ztrátě uhlíku a ze dvou molekul poměrně silné kyseliny mléčné vzniká jedna molekula slabší kyseliny máselné.

3.2.5 Ostatní mikroorganismy

Octové bakterie, plísně, bacily a listerie se vyskytují v silážích v různé míře. Všechny tyto mikroorganismy vesměs do anaerobních procesů v siláži nezasahují a množí se, pokud není siláž dostatečně utěsněna nebo po otevření. Růst je omezen na povrchovou vrstvu do 20 cm a dochází k tvorbě toxických látek a ztrátě kvality. [4]

4. Silážní aditiva

Do silážované suroviny je za určitých okolností vhodné vložit určité látky, které se v podstatě snaží vyrovnat chybějící faktory pro silážování např. bakterie mléčného kvašení a zkvasitelné cukry, nebo se snaží zabránit zkažení siláže (konzervační látky). Jako silážní přísady se používají:

- **Aditiva redukující kažení siláže** (kyselina propionová, octová, mravenčí a sorbová, allicin)
- **Aditiva zvyšující obsah dusíku** (močovina)
- **Aditiva zvyšující obsah zkvasitelných cukrů** (melasa, hydrolytické enzymy)
- **Aditiva bakterií mléčného kvašení** (*Lactobacillus plantarum, L. buchneri, Enterococcus faecium* a ostatní)

Z uvedeného třídění je zřejmé, že jednotlivá aditiva se liší nejen chemickým složením, ale také jejich účinnou látkou, mechanismem účinku a rychlostí, s jakou snižují pH a tvorbu fermentačních kyselin popř. nežádoucí mikroflóru. [4]

Úkolem chemických aditiv (inhibitorů fermentace) je pomocí rychlého snížení hodnoty pH snižovat aktivitu nežádoucí skupiny mikroorganismů a omezit tak tvorbu nežádoucích fermentačních produktů. Tyto produkty jsou úspěšně využívány zejména při silážování rostlinného materiálu s vyšší vlhkostí a nižším obsahem lehce rozpustných cukrů.

Aditiva na biologické bázi se používají ke stimulaci fermentačního procesu. Nelze je použít za nepříznivého počasí při nízkém obsahu sušiny (26 – 28 %), potom zpravidla nezabrání degradaci bílkovin, včetně vysoké tvorby kyseliny máselné, při současné vysoké hodnotě pH, protože k regulaci fermentačního procesu chybí dostatek vodorozpustných cukrů. Přednostmi těchto aditiv jsou především jejich zdravotní nezávadnost, ekologičnost, snížení ztrát sekundární fermentací, zlepšení chutnosti a stravitelnosti. [3]

Silážní přídavky nejsou nezbytně nutné, rozsah použití se mění od země k zemi, přičemž závisí na různých typech silážních technologií, zeměpisných a klimatických podmínkách a ekonomické situaci.

5. Hodnocení siláže

Zcela jistě nejlepším znakem kvalitní siláže je následná efektivní produkce mléka a dobré přírůstky živé váhy krmených zvířat. Protože je ale produkční užitkovost hospodářských zvířat výsledkem mnoha dalších faktorů, je třeba kvalitu siláže hodnotit pomocí organoleptických, nebo lépe rutinních laboratorních testů. Mezi doporučené laboratorní analýzy patří:

- **Stanovení sušiny** – kde platí, že nižší obsah vede častěji k sekundárnímu kvašení, zatímco příliš vysoký obsah sušiny bývá spojován s náchylností k plesnivění.
- **pH** – dobře fermentovaná siláž má pH okolo 4. Obecně u dobře konzervované siláže platí, že čím vyšší sušina, tím vyšší pH.
- **Stanovení kyselin a alkoholů** – hlavní je vysoký obsah laktátu, vysoký obsah etanolu má za následek horší aerobní stabilitu.
- **Stanovení stravitelnosti a energetické hodnoty** – stravitelnost lze odvodit ze stanovení ligninu a vlákniny. Energetická hodnota se vyjadřuje jako metabolizovatelná energie. Obecně jsou její hodnoty u siláže nižší než u obilovin, ale vyšší než u čerstvé píce a sena.
- **Organoleptické hodnocení** – subjektivní stanovení barvy, textury, vůně a chuti.

Složení ideální siláže zobrazuje následující tabulka:

Tabulka 3: Laboratorní analýza ideální siláže [4]

Parametr	**Ideální hodnota**
Sušina (g/kg)	300 – 350
pH	4,0 – 4,2
Popeloviny (g/kg sušiny)	< 80
Hrubý protein (g/kg sušiny)	150 – 170
Kyselina mléčná (g/kg sušiny)	100 – 150
Kyselina octová (g/kg sušiny)	20 – 30
Kyselina máselná (g/kg sušiny)	0
Etanol (g/kg sušiny)	< 10
ME (MJ/kg sušiny)	> 11
Amonný dusík (g/kg celkového dusíku)	< 50
Aminokyselinový dusík (g/kg celkového rozpustného dusíku)	> 700

6. Hodnocení aerobní stability

Podstatou sekundární fermentace (aerobní nestability) je zvýšená aktivita plísní a kvasinek. Tyto mikroorganismy rozkládají organickou hmotu, čímž zvyšují ztráty živin, ale také produkují jedovaté látky. Hlavním problémem, jak zabezpečit stabilitu konzervovaného krmiva, je potlačit činnost těchto mikroorganismů.

Důvodů, proč k nestabilitě dochází, je několik. V prvé řadě se zvýšila sklizňová sušina z 30 % na 40 % přitom je optimální sušina jak z hlediska výnosu organických živin, tak i z hlediska fermentačního procesu v rozmezí 28 až 45 %. Dále má vliv nedostatečný odběr ze žlabů a vaků, kdy odebíraná vrstva není dostatečná a část siláže je vystavena vzduchu po dlouhou dobu. Dalším problémem je nedostatečné zakrývání silážního materiálu a s tím spojené sekundární kvašení. Následkem toho je zahřívání a aerobní nestabilita siláží.

Aerobní stabilita se tak stala důležitým kriteriem k posouzení fermentace a úspěchu konzerva krmiv a tím hodnocení případných ztrát. [11]

6.1 Hodnocení aerobní stability v laboratoři

V laboratorních podmínkách se uplatňují tři základní metody pro hodnocení ztrát vznikajících sekundární fermentací:

1. Stanovení produkce CO_2 buď absorpcí, nebo speciálními analyzátory na CO_2. Toto měření je účinné a umožňuje stanovit ztráty sušiny, které jsou určeny produkcí CO_2 jednotlivými mikroorganismy respirační cestou.
2. Měřením spotřeby O_2 ve Spromat systému speciálně konstruovanému pro stanovení potřeb O_2 odpadní vody. Zde lze znovu přepočítat na ztráty sušiny.
3. Stanovení stoupající teploty jako aerobních ztrát, protože mineralizační procesy jsou exotermické. V objemných vzorcích vzniká akumulace tepla kvůli izolačním účinkům píce. Pokud jsou použity malé vzorky je nutná dodatečná izolace, aby se snížila tepelná výměna s atmosférou. Při standardizaci podmínek je možné přepočítat teplotu na ztráty sušiny. [5],[9]

Protože jsou první dvě uvedené metody komplikované a značně nákladné, začala se poměrně jednoduchá metoda zaznamenávání teploty uvedená v bodě 3 často používat pro hodnocení aerobní stability fermentovaných krmiv. Tento systém je použitelný pro velký počet vzorků, a proto je brán jako standardní metoda.

6.2 Měření teploty jako indikátoru aerobní nestability

Tento systém je znám jako Völkenrode systém pro určení nestability. Obrázek 4 popisuje základ této metody. Jde o lineární korelaci mezi vzrůstem teploty a produkcí CO_2 nebo ztrát sušiny. Tato data byla získána simultánním měřením teplot vzorků a produkce CO_2.

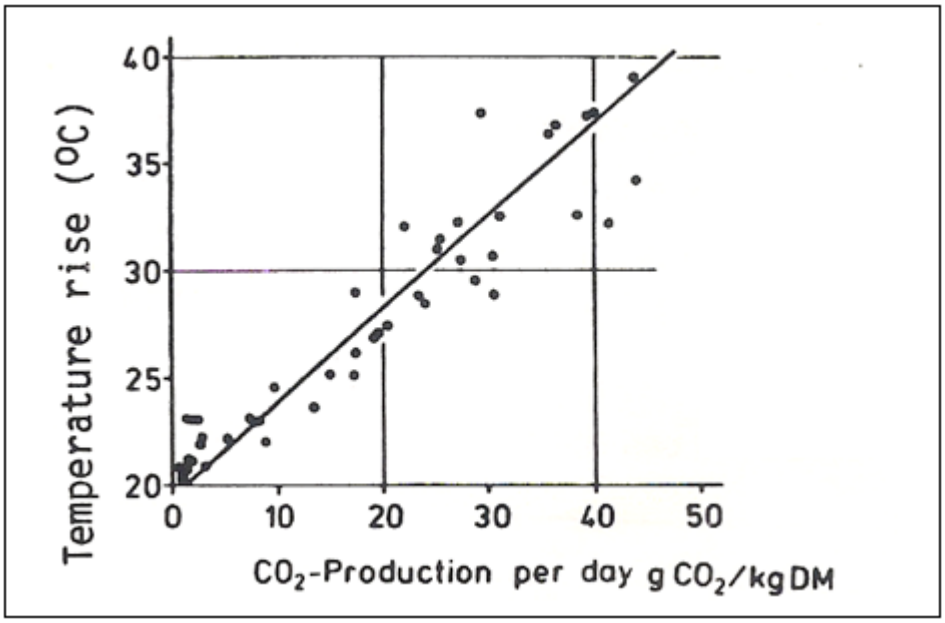
Měření probíhá v malých kontejnerech, do kterých se umístí vzorek krmiva. Tyto kontejnery jsou uloženy v polystyrenových blocích s kapsami na kontejnery. Teplota vzorku je potom měřena v geometrickém středu vpichovými senzory. Kontejner samozřejmě musí mít otvor, aby docházelo k výměně vzduchu. Obrázek 5 zobrazuje umístění kontejnerů se vzorky v polystyrenovém bloku, aby nedocházelo k vyrovnávání teplot mezi vnitřním a vnějším prostředím vzorku. Celý blok se vzorky je potom umístěn v místnosti s relativně stálou teplotou okolí (např. 20°C). Tato teplota okolí bývá nazývána ambient.

Metoda Völkenrode se potom nejčastěji vyhodnocuje jako:

- Doba (ve dnech, případně hodinách), za kterou teplota vzorku vzrostla o 3°C nad okolní teplotu.
- Celkový součet teplot, přičemž teploty je nutné zaznamenávat minimálně 2x denně.

Nově se ve světě začíná také hodnotit podle metody O'Kiely. Referenční teplota se měří ve vodě ve stejném typu boxu, jako se měří teplota siláže. Měření trvá maximálně osm dnů při teplotě vody zhruba 25 °C. Vyhodnocuje se jako:

- Doba, za kterou teplota vzorku vzroste o 2 °C nad okolní teplotou.
- Maximální teplota.
- Celkový součet teplot nad tu referenční za prvních pět dnů. [2]



Obrázek 4: Změna teploty v závislosti na množství CO_2 [8]

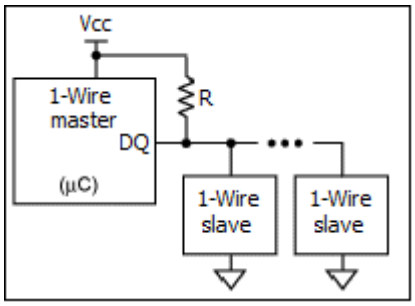
Obrázek 5: Umístění kontejnerů se vzorky

7. Měření teploty čidlem DS18S20

V závěrečné práci jsou použita teplotní čidla od firmy Dallas Semiconducor s označením DS18S20. Tato čidla mají široké použití v průmyslových systémech snímání, řízení a regulace teploty. Široké použití plyne z jejich výhodných vlastností a možnosti zapojení pomocí tzv. 1-wire technologie. Vlastnosti čidla a sběrnice jsou popsány níže.

7.1 1-wire technologie

Tato technologie má český ekvivalent nejspíš jako 1-vodičová sběrnice. Byla navržena firmou Dallas Semiconductor. Základem 1-wire technologie je sériový protokol, který ke komunikaci používá jeden datový vodič a referenční zem. Potom jeden řídící obvod (master) iniciuje a řídí komunikaci s jedním či více obvody ovládaných (slave) na 1-wire sběrnici. Každé slave zařízení má své jedinečné neměnné továrně naprogramované 64-bitové identifikační číslo (ID), které slouží jako adresa zařízení na sběrnici. Typické 1-wire slave zařízení pracuje v rozsahu napětí 2,8 V až 5,25 V. Zařízení nemusí obsahovat ani napájecí PIN, protože je napájeno skrz 1-wire sběrnici. Takovéto napájení je označováno jako parazitní. Zapojení této technologie popisuje obrázek 6. 1-wire technologie je jediná, která používá dva kontakty, data a zem pro obousměrnou komunikaci. Odpojení od sběrnice nebo ztráta kontaktu uvede slave zařízení do definovaného reset stavu. Když se slave připojí nebo se obnoví napájení, oznámí svoji přítomnost na sběrnici. [9]



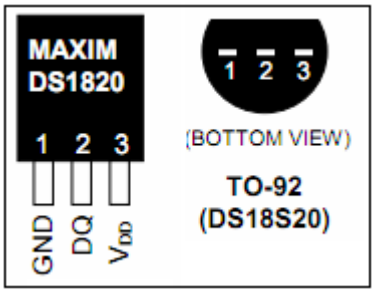
Obrázek 6: Popis 1-wire sběrnice [9]

7.2 Teplotní čidlo DS18S20

Čidla jsou vyrobena také firmou Dallas Semiconductor a proto je vhodné jejich použití na 1-wire sběrnici. Jsou to senzory s přímým digitálním výstupem. Splňují hlavní předpoklady pro fungování na této technologii. Mají rozsah provozních teplot -55 až +125°C s přesností ±0,5°C v rozsahu -10 až +85°C. Kromě toho lze tato čidla napájet přímo z datové linky v tzv. „parazitním režimu“, tudíž bez potřeby externího napájení. Každé DS18S20 má unikátní 64bitové sériové číslo, které umožňuje fungování více těchto čidel na jedné 1-wire

síti. Proto je vhodné použití jednoho mikroprocesoru pro ovládání mnoha čidel distribuovaných na rozlehlé oblasti. Aplikace, které mohou těžit z této funkce, jsou pak: systémy monitorující teploty uvnitř budov, vybavení nebo zařízení a monitorování různých termických procesů a kontrolních systémů. [6]

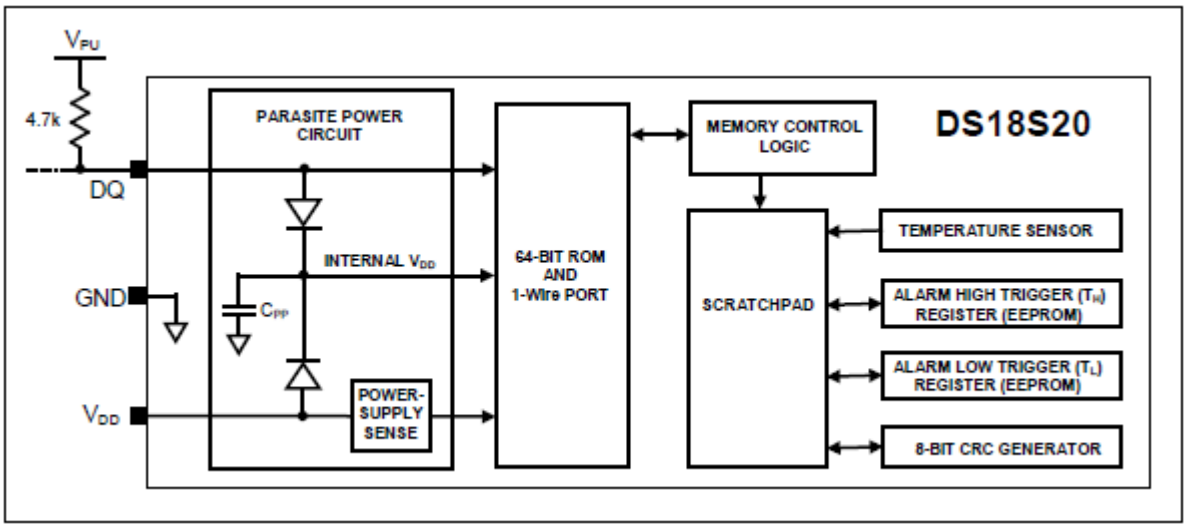
7.2.1 Popis čidla



Obrázek 7: Popis pinů teplotního čidla DS18S20 [9]

Obrázek 7 popisuje vývody a pouzdro senzoru. Tyto jsou dodávány v pouzdře T0-92, ze kterého vedou tři vývody (piny). Použití tohoto pouzdra je výhodné vzhledem k jeho rozměrům. Pin 1 představuje zem a je označován jako GND. Pin 2 je označen DQ a představuje datový vstup/výstup, ale také přivádí k zařízení napětí, pokud je zapojeno v parazitním režimu napájení. Poslední pin je označován jako V_{DD} a přestavuje pin pro napájení. Při parazitním napájení je tento pin spojen s pinem GND.

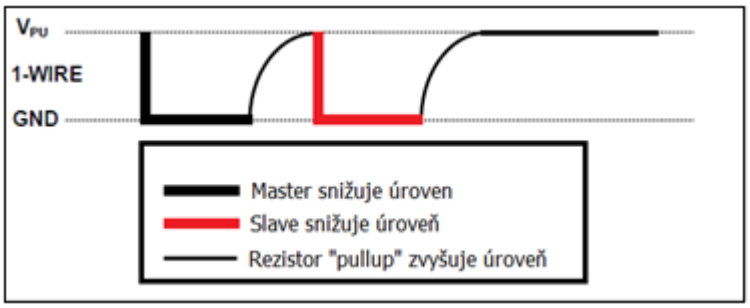
Nejjednodušší popis vnitřního uspořádání senzoru je pak pomocí blokového diagramu. 64bitová ROM paměť obsahuje unikátní sériové číslo. Vnitřní SCRATCHPAD paměť obsahuje 2bajtový teplotní registr, který uchovává digitální výstup z teplotního čidla. Kromě toho SCRATCHPAD poskytuje přístup k 1bajtovému hornímu a dolnímu registru prahu alarmu (TH a TL). TH a TL jsou typu EEPROM, takže uchovávají údaje, i když je zařízení vypnuté. Blokový diagram je na obrázku níže.



Obrázek 8: Blokový diagram vnitřního uspořádání senzoru DS18S20 [6]

Využívá se unikátního 1-wire protokolu, který implementuje komunikaci pro sběrnici pomocí jednoho řídícího signálu. Kontrolní linka vyžaduje slabý pullup rezistor, protože všechna zařízení jsou napojena přes budič s otevřeným kolektorem (přes DQ). V tomto systému, mikroprocesor (master) identifikuje a sbírá data ze zařízení na sběrnici pomocí jejich unikátní 64bitové adresy. Protože každé čidlo má svoji unikátní adresu, lze jich teoreticky na sběrnici použit neomezené množství. [6]

Dalším rysem DS18S20 je schopnost pracovat bez externího napájení. Místo toho je energie dodávána přes pul lup rezistor skrz pin DQ když je sběrnice aktivní ve stavu logické 1. Vysoká úroveň sběrnice V_{Pu} také nabíjí vnitřní kondenzátor CPP, který pak napájí zařízení, když je sběrnice uzemněná (na nízké úrovni napětí). Obrázek 9 popisuje tyto stavy.



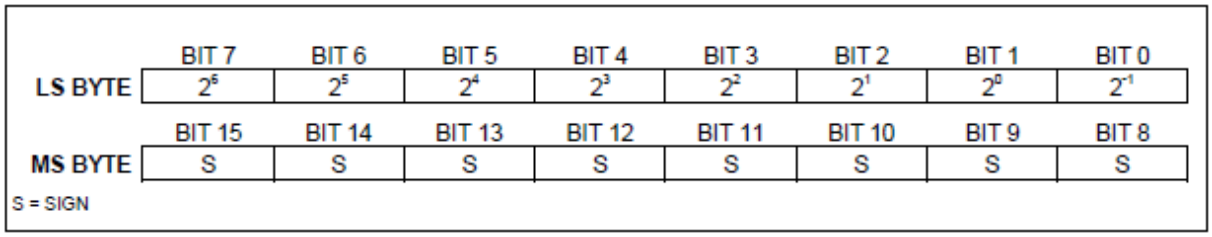
Obrázek 9: Popis stavů na 1-wire sběrnici [9]

Tato metoda, kdy je energie dodávána skrz 1-wire síť, je označována jako parazitní, napájení. Míst toho může být čidlo napájeno externě přes pin Vdd.

7.2.2 Měření teploty DS18S20

Senzory DS18S20 zobrazují přímý digitální výstup tzv. direkt-to-digital. Výstup čidla má 9-bitové rozlišení, které odpovídá 0,5°C krokům. K zahájení měření teploty a A-D

převodu je nutné aby master provedl T(44h) příkaz. Po převodu jsou výsledná teplotní data uložena v 2-bajtovém registru v zápisové paměti (scratchpad memory) a čidlo se vrátí do klidového stavu.



Obrázek 10: Uspořádání teplotního registru [9]

Výstup z čidel je kalibrován ve stupních celsia. Údaj o teplotě je v teplotním registru uchováván jako 16-bitové číslo. Obrázek 10 zobrazuje uspořádání teplotního registru. Znakové bity S udávají, je-li teplota pozitivní nebo negativní: pro pozitivní je S=0 pro negativní je S=1. Tabulka 4 ukazuje příklady digitálních výstupů dat a dopovídajících teplot. [6]

Tabulka 4: Digitální výstup dat a odpovídající teploty

Teplota (°C)	**Digitální výstup (binárně)**	**Digitální výstup (hexadecimálně)**
+85,0	0000 0000 1010 1010	00AAh
+25,0	0000 0000 0011 0010	0032h
+0,5	0000 0000 0000 0001	0001h
0	0000 0000 0000 0000	0000h
-0,5	1111 1111 1111 1111	FFFFh
-0,25	1111 1111 1100 1110	FFCEh
-0,55	1111 1111 1001 0010	FF92h

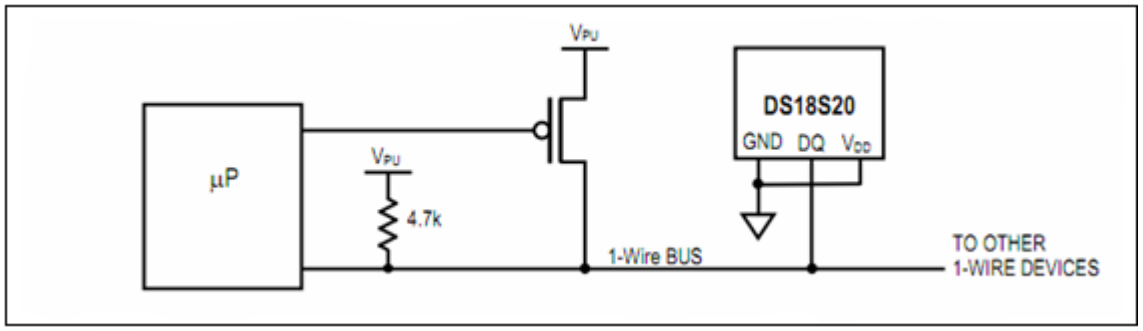
Rozlišení větší než 9 bitů je možné vypočítat na základě dat COUNT REMAIN a COUNT PER °C registrovaných v zápisníku (scratchpad registru). Všimněme si, že COUNT PER °C registr je přímo připojen na 16 (10h). Po přečtení zápisníku je hodnota TEMP_READ získána ořezáním 0,5°C bitu (bit 0) z teplotních dat viz obrázek 10. Rozšířené rozlišení teploty pak lze vypočítat pomocí následující rovnice:

$$TEMPERATURE = TEMP\_READ - 0{,}25 + \frac{COUNT\_PER\_C - COUN\_REMAIN}{COUNT\_PER\_C} \qquad (7)$$

7.2.3 Napájení čidel

Čidla mohou být napájena z externího zdroje na Vdd pinu, nebo mohou pracovat v parazitním režimu, který jim umožňuje fungovat bez místního externího napájení. Pro naše účely bude použito pouze parazitní napájení.

V režimu parazitního napájení může 1-wire sběrnice a CPP poskytnout dostatek proudu pro čidlo a většinu jeho operací tak dlouho, dokud je splněno specifické časování a požadované napětí. Nicméně když čidlo provádí teplotní konverzi nebo kopírování dat z paměti EEPROM do zápisníku, může být provozní proud až 1,5 mA. Tento proud může způsobit nepřijatelný úbytek napětí na slabém 1-wire pullup rezistoru, větší než může být nahrazen kondenzátorem CPP. Aby bylo zajištěno, že má čidlo dostatek proudu, je nezbytné zajistit dostatečně silný pullup na sběrnici, kdykoliv probíhá převod teploty nebo se kopírují data ze zápisníku do EEPROM. Toho lze dosáhnout pomocí polem řízeného tranzistoru MOSFETu přímo na 1-wire sběrnici, jak je uvedeno na obrázku11.



Obrázek 11: Schéma parazitního napájení [6]

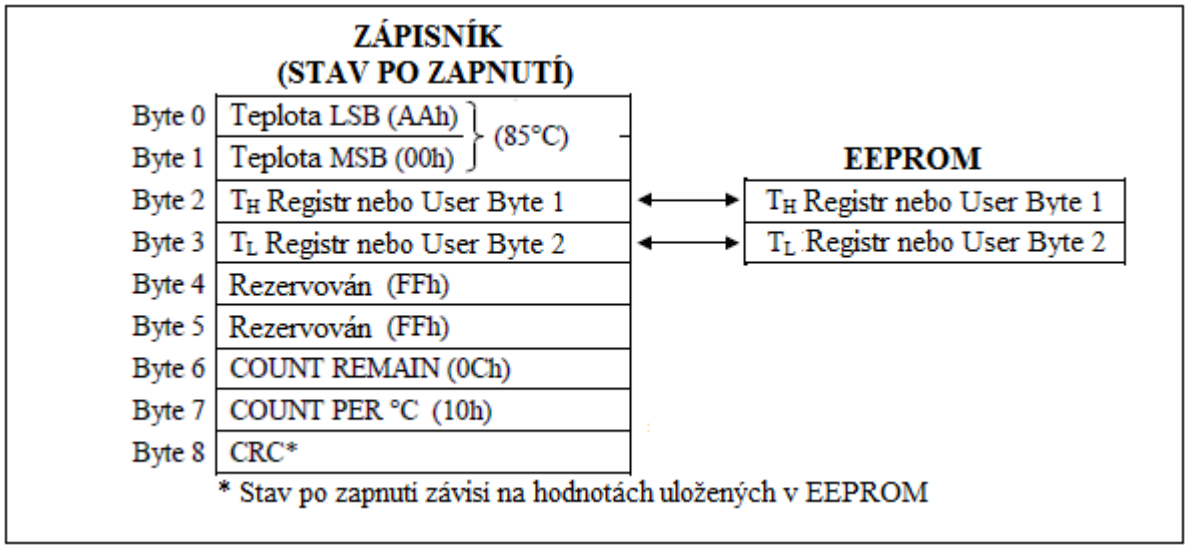
Sběrnice musí být nastavena na vysokou úroveň pollup na dobu 10µs (max) po Convert T[44h] nebo Copy Scratchpad [48h] příkazech, a také po celou dobu konverse (tCONV) nebo přenosu dat (tWR=10 ms). Žádná jiná aktivita se nemůže konat na sběrnici, zatím co je pullup aktivní.

Použití parazitního napájení se nedoporučuje při měření teplot nad 100 °C, protože čidla nemusí být schopna udržet komunikaci kvůli vyšším svodovým proudům, které mohou existovat při těchto teplotách. V takových aplikacích se doporučuje napájení externím zdrojem.

V některých situacích nemusí master vědět, zda je čidlo na sběrnici napájeno parazitně nebo z vnějších zdrojů. Master tyto informace potřebuje, aby zjistil, jestli je potřeba vysoká úroveň pullup během konverse teploty. Chce-li získat tyto informace, vydá příkaz Skip ROM [CCh] následovaný Read Power suply [B4h] příkazem, který je následován „read time slot“. Během read-time slot, sníží čidlo úroveň napětí sběrnice. Externě napájené čidlo by nechalo sběrnici na vysoké úrovni, zatímco když zůstane úroveň snížena, master zjistí, že musí poskytnout silný pullup na sběrnici během konverse teploty. [6]

7.2.4 Paměť čidel DS18S20

Paměť čidel je organizována podle obrázku 12. Skládá se ze zápisníku SRAM a permanentní paměti EEPROM pro uložení alarmů TH a TL. V případě, že nejsou použity alarmy, pak může TH a TL registr sloužit jako universální paměť.



Obrázek 12: Uspořádání paměti DS18S20 [6]

Bajt 0 a bajt 1 na zápisníku obsahuje LSB a MSB z teplotního registru. Tyto bajty jsou jen pro čtení. Bajty 2 a 3 umožňují přístup k TH a TL registrům. Bajty 4 a 5 jsou vyhrazeny pro vnitřní použití zařízení a nemohou být přepsány, vrátí samé 1, pokud jsou čteny. Bajty 6 a 7 obsahují COUNT REMAIN a COUNT PER C registry, které mohou být použity pro počítání vyššího rozlišení měření teploty, jak již bylo vysvětleno výše. Bajt 8 v zápisníku je jen pro čtení a obsahuje CRC kód pro bajty 0 až 7 zápisníku. Čidlo generuje tento CRC kód použitím speciální metody. [6]

Další data jsou zapsána do bajtů 2 a 3 zápisníku pomocí příkazu *Write scratchpad* [4Eh], data musí být odeslána do DS18S20 počínaje nejméně signifikantním bitem z bajtu 2. Aby byla ověřena integrita dat, je možné číst zápisník (pomocí příkazu *Read Scratchpad* [BEh]) jakmile jsou data již zapsána. Když je čten zápisník, data jsou přenášena přes 1wire sběrnici počínaje nejméně významným bitem z bajtu 0. Aby byla přesunuta TH a TL data ze zápisníku do EEPROM paměti, master musí zadat příkaz *Copy Scratchpad* [48h].

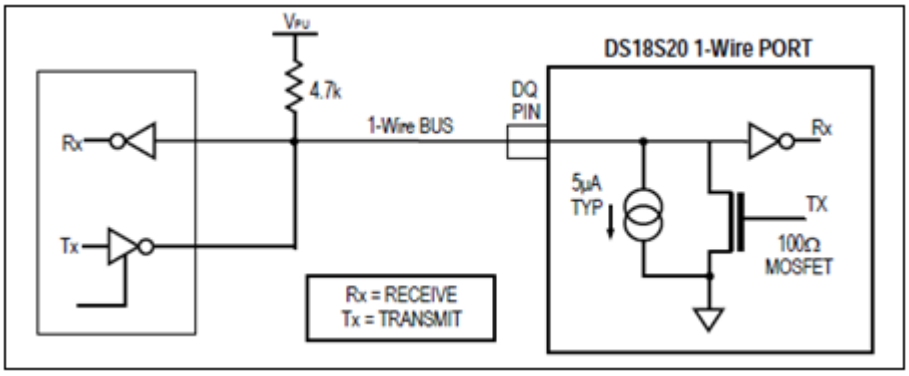
Data v paměti EEPROM jsou dostupná, pokud je přístroj vypnutý, po zapnutí jsou EEPROM data překopírována do odpovídajícího místa v zápisníku. Data mohu být také překopírována z EEPROM paměti do zápisníku kdykoliv po použití příkazu Recall E2 [B8h]. Master pošle příkaz “read time slots” následován Recall E2 příkazem a čidlo indikuje probíhající recall jako 0 a dokončení recallu jako 1.

8. 1-wire sběrnice pro čidla DS18S20

Tento systém používá jednoho masteru k řízení více slave zařízení. DS18S20 je vždy jako řízené (slave). Pokud je jedno řízené zařízení, systém je označován jako „single drop“, zatímco existuje-li více řízených je systém označován jako „multi drop“. Všechna data a příkazy jsou přenášena jako nejméně významné bity po 1-wire sběrnici. Následující text o 1-wire sběrnici pro tato čidla je rozdělen do tří tematických okruhů: hardwarová konfigurace, transakční sekvence a 1-wire signalizování (typy signálů a časování).

8.1 Hardwarová konfigurace

1-wire sběrnice má samozřejmě jen jednu datovou linku. Každé zařízení (master a slave) vstupuje na datovou linku přes otevřený kolektor nebo přes open drain. To umožňuje každému zařízení „uvolnit/vypustit“ datovou linku pokud zařízení nepřímá data, takže sběrnice je přístupná pro další zařízení. 1-wire port DS18S20 (DQ pin) je otevřeným kolektorem s vnitřním okruhem ekvivalentním k tomu na obrázku 13.



Obrázek 13: Vnitřní okruh – hardwarová konfigurace [6]

1-wire sběrnice vyžaduje externí pullup rezistor přibližně 5kΩ, proto je v klidovém stavu úroveň napětí sběrnice vysoká. Pokud musí být z nějakého důvodu operace přerušeny, musí zůstat sběrnice v klidovém stavu, aby mohly operace zase pokračovat. Doba zotavení mezi bity může trvat tak dlouho, dokud je sběrnice neaktivní během zotavování. Pokud je sběrnice držena v nečinnosti déle než 480µs, všechny komponenty na sběrnici jsou resetovány. [6]

8.2 Transakční sekvence

Transakční sekvence pro přístup k DS18S20 je následující:

1. inicializace
2. ROM příkazy (následují všechny potřebné výměny dat)

3. DS18S20 funkční příkazy (po jakékoliv požadované výměně dat)

Je velmi důležité dodržovat toto pořadí při každém přístupu na DS18S20, protože čidlo neodpoví, pokud nějaké kroky v pořadí chybí nebo jsou ve špatném pořadí. Výjimkou z těchto pravidel je příkaz *search ROM* a *Alarm Search*. Po vydání některého z těchto příkazů se musí master vrátit ke kroku 1.

8.2.1 Inicializace

Všechny přenosy na 1-wire sběrnici začínají inicializační sekvencí. Ta se skládá z reset impulsu, přenášeného masterem, následovaným presenčním pulsem vysílaného slave zařízením. Presenční puls umožňuje řídícímu zařízení zjistit, zda jsou na sběrnici připojena slave zařízení a zda jsou připravena k provozu. Časování těchto pulsů je detailně popsáno v sekci 1-wire signalizace.

8.2.2 Příkazy ROM

Jakmile řídící zařízeni (master) detekuje presenční puls, může vyslat příkaz ROM. Tyto příkazy pracují na unikátním 64bitovém kódu jednotlivých slave zařízení a umožňují master zařízení ovládat právě jedno ze zařízení přítomných na lince. Tyto příkazy umožňují také zjistit kolik zařízení a jakého typu je přítomno na sběrnici, anebo zda některé nepřekročilo mez alarmových stavů. Je pět příkazů a každý je 8bitový. Master zařízení musí vykonat příslušné ROM příkazy ještě před vykonáním funkčních příkazů. Názvy příkazů jsou ponechány v angličtině.

1. **Search ROM [F0h]**

Když je systém spuštěn poprvé, musí master identifikovat ROM kódy všech slave zařízení na lince, což umožňuje masteru určit počet a typy jednotlivých přítomných zařízení. Master se naučí ROM kódy přes proces eliminace, který vyžaduje aby master provedl Search ROM cyklus (tj. Search ROM příkaz následovaný výměnou dat) tolikrát, kolikrát je nezbytné k identifikaci všech zařízení. Po každém cyklu Search ROM se musí master vrátit ke kroku 1 (inicializace) v transakční sekvenci.

2. **Read ROM [33h]**

Tento příkaz lze použít, pouze pokud je na sběrnici jen jedno slave zařízení. Umožňuje masteru přečíst jeho 64bitový kód bez použití Search ROM postupu. Pokud se tento příkaz použije, je-li na lince více slave zařízení, dojde ke kolizi, pokud se budou snažit slave zařízení operovat ve stejnou chvíli.

3. **Match ROM [55h]**

Příkaz Match Rom následovaný 64bitovým kódem ROM umožňuje řídícímu zařízení řešit konkrétní slave zaříení na sběrnici. Pouze ten slave, který přesně odpovídá 64bitové

ROM kódové sekvenci, bude reagovat na povely a všechna ostatní slave zařízení čekají na reset puls.

4. **Skip ROM [CCh]**

Master může použit tento příkaz k adresování všech zařízení současně bez zaslání jakékoli informace o ROM kódu. Například, master může všechna čidla na sběrnici nechat vykonávat konverzi teploty použitím příkazu Skip ROM následovaným příkazem Convert T[44h].

5. **Alarm SEARCH [ECh]**

Provoz tohoto příkazu je totožný s provozem příkazu Search ROM, kromě toho, že odpoví pouze ta slave zařízení, která mají nastaven alarm. Tento příkaz umožňuje master zařízení zjistit, zda došlo k překročení nastavených limitů na nějakém z čidel.

8.2.3 DS18S20 funkční příkazy

Když master použil ROM příkazy k adresování zařízení, se kterými chce komunikovat, může vydat jeden z funkčních příkazů. Tyto příkazy umožňují řídícímu zařízení psát a číst ze zápisníkové paměti čidel, zahájit konverzi teploty a určit způsob napájení čidla.

1. **Convert T [44h]**

Tento příkaz inicializuje jednu teplotní konverzi. Po převodu jsou výsledná teplotní data uložena v 2-bajtovém teplotním registru zápisníkové paměti a čidlo se vrátí do klidového stavu. Je-li použit parasitní mód napájení, během 10µs (max) poté co je tento příkaz vydán master musí zvýšit pullup na sběrnici po dobu převodu (tCONV), jak je popsáno v části napájení DS18S20. [6]

2. **Write scratchpad [4Eh]**

Tento příkaz umožňuje masteru psát 2-bajty dat do zápisníku DS18S20. První bajt je zapisován do registru TH (bajt 2 v zápisníku), a druhý bajt je zapsán do rejstříku TL (bajt 3). Data musí být předána s nejméně významným bitem na prvním místě. Oba bajty musí být zapsány dříve, než master odešle *reset pulse*, jinak může dojít k poškození dat.

3. **Read scratchpad [BEh]**

Tento příkaz umožňuje masteru číst obsah zápisníku. Přenos dat začíná nejméně významným bitem bajtu 0 a pokračuje čtení přes zápisník až do 9. bajtu (bajt 8 CRC). Master může vyslat *reset*, aby ukončil čtení kdykoliv, když je potřeba pouze část zápisníku.

4. **Copy scratchpad [48h]**

Tento příkaz kopíruje obsah zápisníku TH a TL registrů (bajty 2 a 3) do paměti EEPROM.

5. **Recall E2 [B8h]**

Tento příkaz připomíná hodnoty alarmů (TH a TL) z EEPROM a umístí data do bajtů 2 a 3, respektive do paměti zápisníku. Master může vydat příkaz read-time slots následován Recall E2 příkazem a čidlo indikuje stav průběhu odvolání jako 0 a jeho ukončení jako 1. Tato funkce je prováděna automaticky při spuštění přístroje, takže platná data jsou dostupná hned, jakmile je zařízeno připojeno k napájení.

6. **Read power supply [B4h]**

Master vysílá tento příkaz následovaný read-time slots, aby zjistil, zda je některé ze zařízení na sběrnici napájeno parasitně. [6]

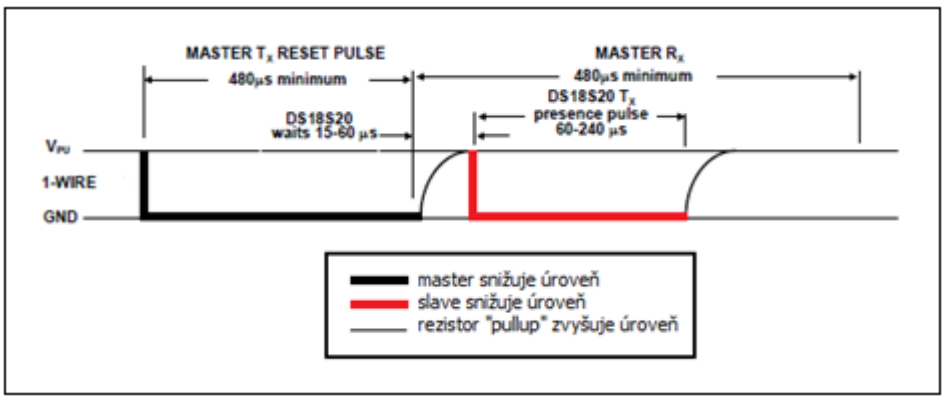
8.3 1-wire signalizace

DS18S20 používá přímý 1-wire komunikační protokol pro zajištění integrity dat. Některé druhy signálu jsou definovány tímto protokolem: reset puls, presenční puls, wrie 0, write 1, read 0 a read 1. Všechny tyto signály s výjimkou presenčního pulsu vysílá master zařízení.

8.3.1 Iniciační procedura – reset a present puls

Veškerá komunikace s DS18S20 začíná inicializační sekvencí, která se skládá z reset pulsu od mastera a presenčního pulsu od DS18S20. Toto je znázorněno na obrázku 14. Když čidlo odešle presenční puls v reakci na reset, master zaznamená, že je na sběrnici a připraveno k použití.

Během inicializační sekvence přenáší master TX reset puls a sníží úroveň na sběrnici na minimálně 480µs. Master potom uvolní hladinu a přejde do režimu příjmu RX. Po uvolnění sběrnice táhne pullup odpor úroveň nahoru. Když čidlo DS18S20 zjistí tuto náběžnou hranu, čeká od 15µs do 60 µs a potom vyšle presenční puls, tak že sníží úroveň na 60 až 240 µs. [6]



Obrázek 14: Inicializační časování [6]

8.3.2 Čtení/zápis časových intervalů

Master zapisuje data do DS18S20 během zápisu časových intervalů a čte data během čtení časových intervalů. Jeden bit dat na 1-wire síti je přenášen během jednoho časového intervalu.

- **Psaní časových intervalů**

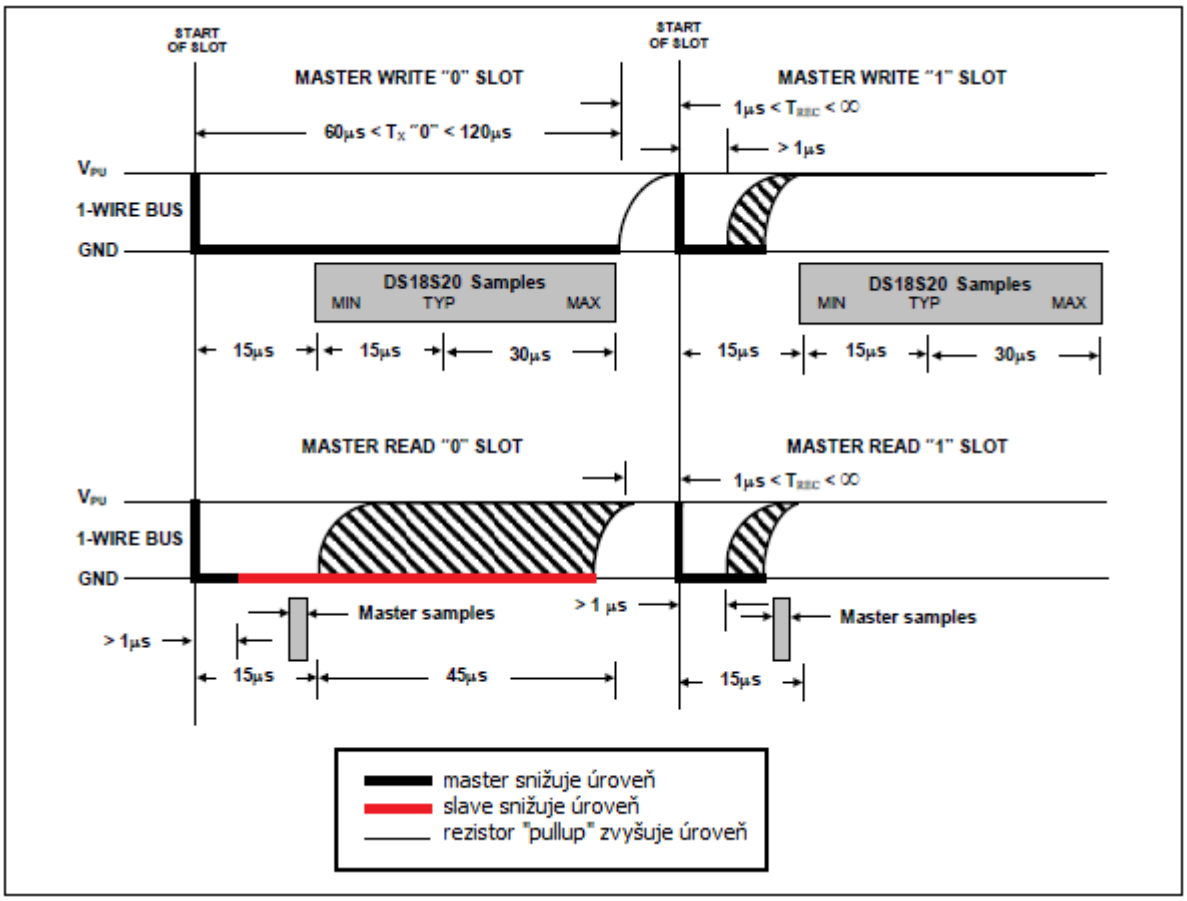
Existují dva typy psaní časových intervalů: Write 1 časové intervaly a Write 0 časové intervaly. Mater používá write 1, aby napsal logickou 1 do DS18S20 a write 0 pro logickou 0. Všechny časové intervaly write musí být dlouhé minimálně 60µs s minimálním časem obnovení 1µs mezi jednotlivými sloty write. Oba typy zahajuje master snížením úrovně 1-wire sběrnice viz obrázek 15.

Aby byl vytvořen write 1 time slot, po snížení úrovně sběrnice, musí master uvolnit sběrnici během 15µs. Když je sběrnice uvolněna, pullup rezistor opět zvýší hladinu. Pro vytvoření write 0 časového intervalu, po snížení úrovně sběrnice musí master držet úroveň nízko po celou dobu časového intervalu (nejméně 60µs). DS18S20 vzorkuje 1-wire sběrnici oknem, s délkou od 15µs do 60µs, potom co master inicializuje psaní časových intervalů. Pokud je úroveň sběrnice vysoká během vzorkování oknem, je zapsána 1 do DS10S20 čidla. Je-li úroveň nízká, pak je zapsána 0 do čidla. [6]

- **Čtení časových intervalů**

DS18S20 může přenášet data na master, pouze když master vyšle příkaz pro čtení časových intervalů. Proto musí master generovat čtení časových intervalů ihned po příkazech *Read Scratchpad* nebo *Read Power Suply*, aby mohlo čidlo poskytnout požadovaná data. Kromě toho může master generovat čtení časových intervalů po příkazech *Convert T* nebo *Recall E2,* aby zjistil stav provozu.

Všechna čtení časových intervalů musí být v minimální délce 60µs s minimálně 1µs zotavení mezi intervaly. Čtení je zahájeno snížením úrovně sběrnice masterem na minimálně 1 µs a uvolněním sběrnice (viz obrázek 15). Poté, co master začne čtení časových úseků, čidlo začne vysílat 1 nebo 0 na sběrnici. Vysílá 1 tak, že nechá úroveň vysoko a 0 naopak snížením úrovně dolů. Když je vysílána 0, čidlo uvolní sběrnici na konci časového intervalu a sběrnice zvýší úroveň pomocí pullup rezistoru. Výstupní data z DS18S20 platí po dobu 15 µs po sestupné hraně, kterou zahájilo čtení časových intervalů. Proto musí master uvolnit sběrnici. [6]



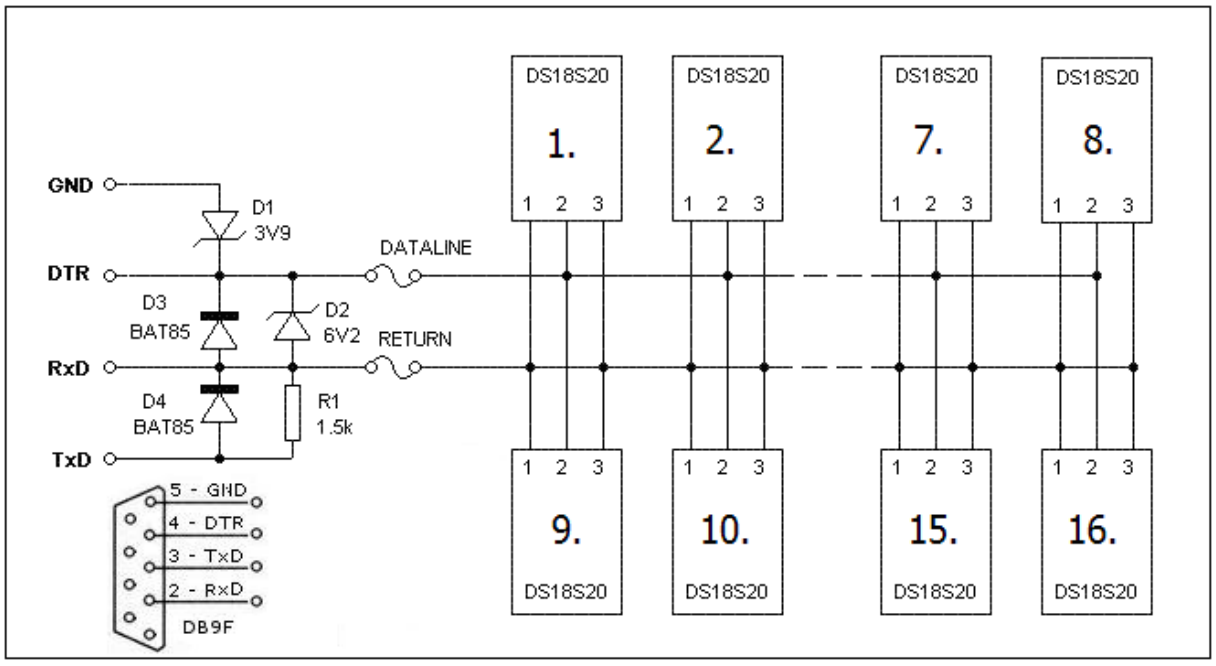
Obrázek 15: Čtení a zápis časových intervalů [6]

9. Připojení 1-wire sítě k PC

Počítač je ideálním prostředkem sloužícím jako master zařízení pro čtení a zápis do prvků na 1-wire sběrnici. Nicméně je potřebný sériový port nebo převodník poskytující dostatečné napětí pro 1-wire technologii. Převodník DS9097E byl vyvinut firmou Dallas Semiconductor jako ekonomické a snadno použitelné rozhraní.

9.1 Převodník DS9097E

Schéma převodníku je znázorněné na obrázku 16. Adaptér je napájen výhradně z COM portu počítače. Převodník umožňuje maximální využití výstupu pouze pomocí několika diod. Zenerova dioda D1 drží datovou linku na 3,9 V, přičemž dioda D2 omezuje maximální napětí na 1-Wire sběrnici na 6,2 V. To také omezuje většinu negativních kolísání napětí na RxD k mínus 2.3 V. Když je TxD pozitivní, Schottkyho dioda D3 omezuje rozdíl napětí mezi 1-Wire data linkou a zpětnou linkou na 0,2 V a D4 připojí TxD na RxD. Tak obchází R1 pullup rezistor a zajišťuje cestu k vytvoření časového intervalu (time slot). Minimální hodnota 1,5 kΩ pullup rezistoru generuje logickou nulu na napětí 0.3 V na sběrnici. Maximální hodnota logické nuly je 0,8 V, což dává rozmezí 0,5 V pro případný šum. [10]

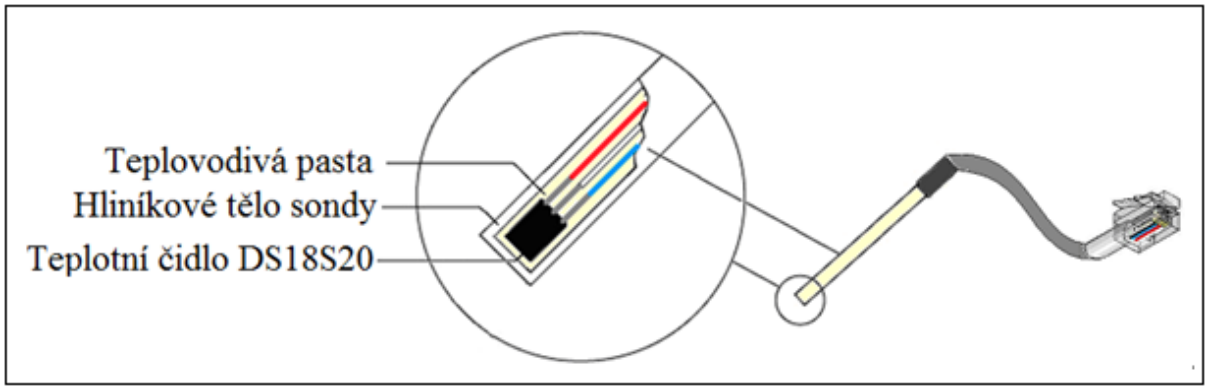


Obrázek 16: Schéma převodníku DS9097E

Převodník je potom ukončen 9-pinovým cannon konektorem, přes který je spojen s počítačem.

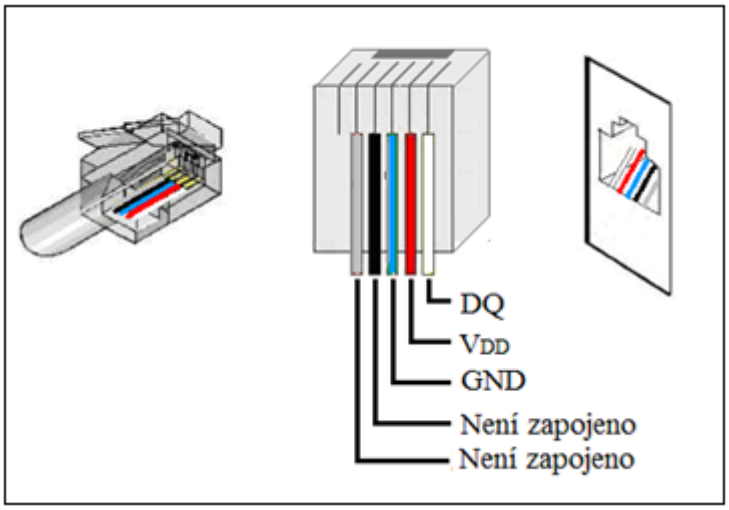
9.2 Vpichové sondy pro měření stability

Pro měření stability vzorků v kontejnerech jsou nejvhodnější vpichové sondy. Jednotlivé senzory jsou umístěny v nerezové trubičce, ze které vede kabel na konci opatřený RJ telefonním konektorem. Průřez takto vyrobeným vpichovým senzorem je zobrazen na obrázku 17.



Obrázek 17: Průřez vyrobené vpichové sondy

Senzory jsou umístěny na koci trubičky a připojeny na vodiče STP kabelu. Volný prostor je vyplněn teplovodivou silikonovou pastou pro zlepšení vedení tepla mezi čidlem a venkovním prostředím. Kabel je na konci opatřen RJ telefonním konektorem. Tyto konektory jsou zde výhodné hlavně pro jejich lehké propojování a snadnou instalaci na kabel. Schéma výstupních pinů konektoru je zobrazeno na obrázku níže. Stínění kabelu a černý vodič v kabelu není zapojen, slouží pouze ke zvýšení mechanické pevnosti konektoru.



Obrázek 18: Nákres vývodů konektoru sondy

10. Programovací prostředí LabVIEW

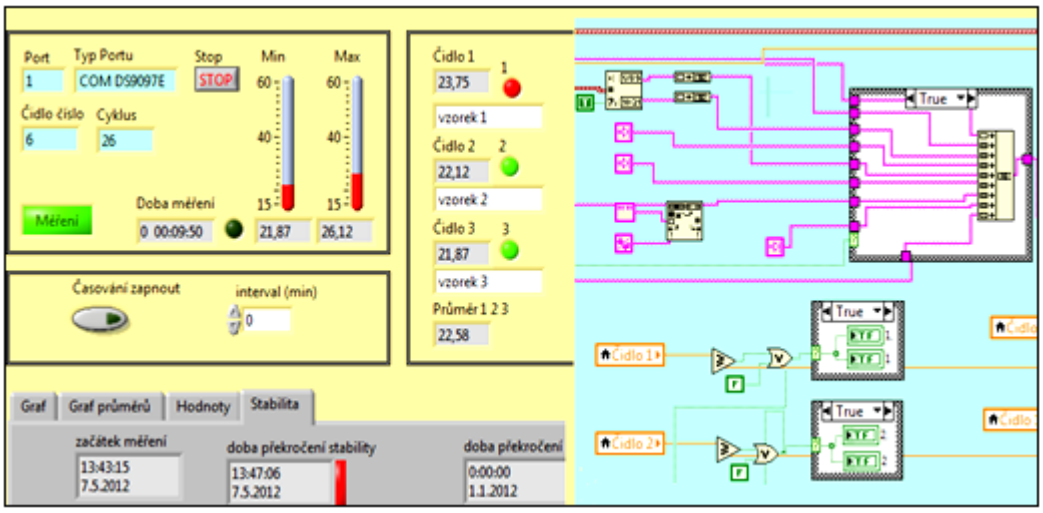
Při vytváření programu bylo použito programovací prostředí LabVEW. Popis funkce a možností tohoto prostředí najdeme níže.

10.1 LabVIEW

Název LabVIEW je zkrácením slov Laboratory Virtual Instruments Engeneering Workbench a toto prostředí bylo vyvinuto americkou firmou National Instruments.

Jde o grafické programovací prostředí, které bývá označováno jako tzv. G jazyk od slova Graphical. Poskytuje veškeré datové i programovací struktury a možnosti, které poskytují klasické textové vývojové prostředí jako je například Visual C++, Matlab atd., ale tvorba programů probíhá pomocí grafických ikon a elementů. Zápis programu je tak podobný tvorbě blokových schémat. Jedná se tedy o nástroj pro teoretická, ale i praktická měření, simulaci a návrh tzv. virtuálních přístrojů s využitím počítačů. Tyto přístroje se v programu nazývají VI, z anglického „virtual instruments“. Takovéto virtuální přístroje jsou svým vzhledem a činností podobné reálným přístrojům, jako např. osciloskopy, multimetry a jiné. Při tvorbě programu vytváříme „Front Panel“ (čelní panel, neboli grafické uživatelské prostředí) pomocí knihovny ovládacích a zobrazovacích prvků. Tyto prvky jsou například tlačítka, rolovací menu, grafy, LED indikátory apod. Samotný algoritmu se vytváří v části „Block Diagram“ (blokový diagram), kde se umisťují a propojují jednotlivé funkce. Výsledný programu pak představuje pomyslný tok signálu mezi jednotlivými funkcemi. Celé prostředí je tak velmi intuitivní a přehledné. [14]

Na obrázku 19 níže jsou vidět jednotlivé struktury. Pro přehlednost je pole čelního panelu podbarveno žlutě, zatím co blokový diagram je podbarven světle modře.



Obrázek 19: Programovací prostředí LabVIEW

11. Popis struktury programu

Při tvorbě celého programu jsme vycházeli ze základních principů a postupů definovaných výrobcem čidel DS18S20.

11.1 Analýza požadavků na program

Pro měření aerobní stability je zapotřebí splnit několik požadavků, tak aby celý výsledný systém vyhovoval požadavkům firmy Nutrivet a správně zobrazoval výsledky měření.

Nejprve je na začátku celého programu potřeba inicializovat port a převodník, na který je připojena 1-wire síť. Dále je zapotřebí identifikovat konkrétní čidlo na 1-wire sběrnici, pomocí jeho specifického ID a provést převod teploty tohoto čidla. Důležitým bodem při měření aerobní stability je taktéž dostatečné teplotní rozlišení, aby nebyla ovlivněna přesnost měření a výsledných hodnot.

Z principů funkce 1-wire sítě vyplývá, že nelze k zařízením na sběrnici přistupovat paralelně ke všem najednou, ale musí se k nim přistupovat jednotlivě. Konverse teploty jednoho čidla trvá o něco déle než jednu sekundu. Z tohoto důvodu trvá jeden cyklus měření, kdy dojde k zobrazení hodnot ze všech šestnácti čidel časový interval přes dvacet sekund.

Měření hodnot z jednotlivých čidel musí probíhat po uběhnutí definovaných časových intervalů. Tyto intervaly by měly být volitelné přímo v uživatelském menu.

Hodnoty čidel musí být zobrazovány do grafů. Pro každé čidlo by měla být zobrazena křivka teploty v závislosti na čase. Všech šestnáct čidel by pak mělo být zobrazeno v jednom grafu. Dále je požadavkem zobrazit průměrné hodnoty z několika čidel a zobrazit je v dalším odděleném grafu.

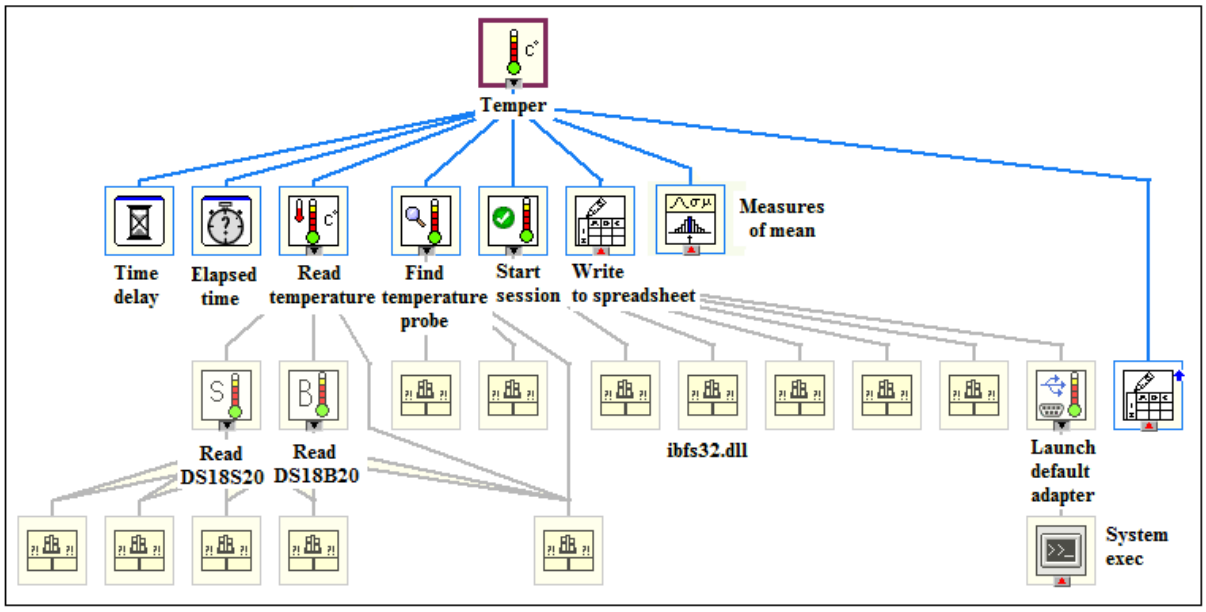
Naměřené hodnoty by pak měly být ukládány do externího souboru typu xls, tak aby při případném pádu systému nebyla naměřená data ztracena. V tomto datovém souboru by se měly ukládat i časové značky jednotlivých měření.

Dále je potřeba stanovit čas začátku měření a čas, kdy teplota konkrétního čidla překročila prahovou hodnotu. Tato je určena jako teplota ambientního čidla zvětšená o tři stupně celsia. Na základě rozdílu těchto dvou časů je poté vypočtena doba stability daného vzorku měřeného konkrétním čidlem. Překročení stability a hodnota stability v hodinách by měla být indikována v uživatelském menu programu.

11.2 Popis hierarchie a souvislostí jednotlivých částí programu

Celý program je řízen přes program Temper.vi. Na něj jsou napojeny jednotlivé podprogramy, které provádějí dílčí úkoly a výpočty.

Aby bylo možné lépe poznat a pochopit souvislosti mezi hlavním programem a jednotlivými podprogramy byl sestrojen blokový diagram níže, který popisuje tyto souvislosti.



Obrázek 20: Blokový diagram hierarchie programu

Z obrázku je patrné, že hlavní program Temper ovládá několik dalších podprogramů. Mezi tyto podprogramy patří *Read temperature*, *Find temperature probe*, *Start session*, *Launch default adapter*, *Read DS18S20* a *Read DS18B20*. Zmíněné podprogramy budou podrobně popsány dále. Potom jsou zde podprogramy přímo z knihovny programu LabVIEW jako je *Time delay*, *Elapsed time*, *System exec* a *Write to spreadsheet*. Poslední je na schématu několikrát zobrazena knihovna *ibfs32.dll*. Tato knihovna obsahuje jednotlivé příkazy, pomocí kterých jsou ovládána čidla DS18S20.

Hlavní program obsahuje podprogram *Start session*, který se odvolává na další podprogram *Launch default adapter*. Dále pak podprogram *Find temperature probe* a nakonec podprogram *Read temperature*. Tento podprogram se odvolává na dva podprogramy *Read DS18S20* a *Read DS18B20*.

11.2.1 Podprogram *StartSession.vi*

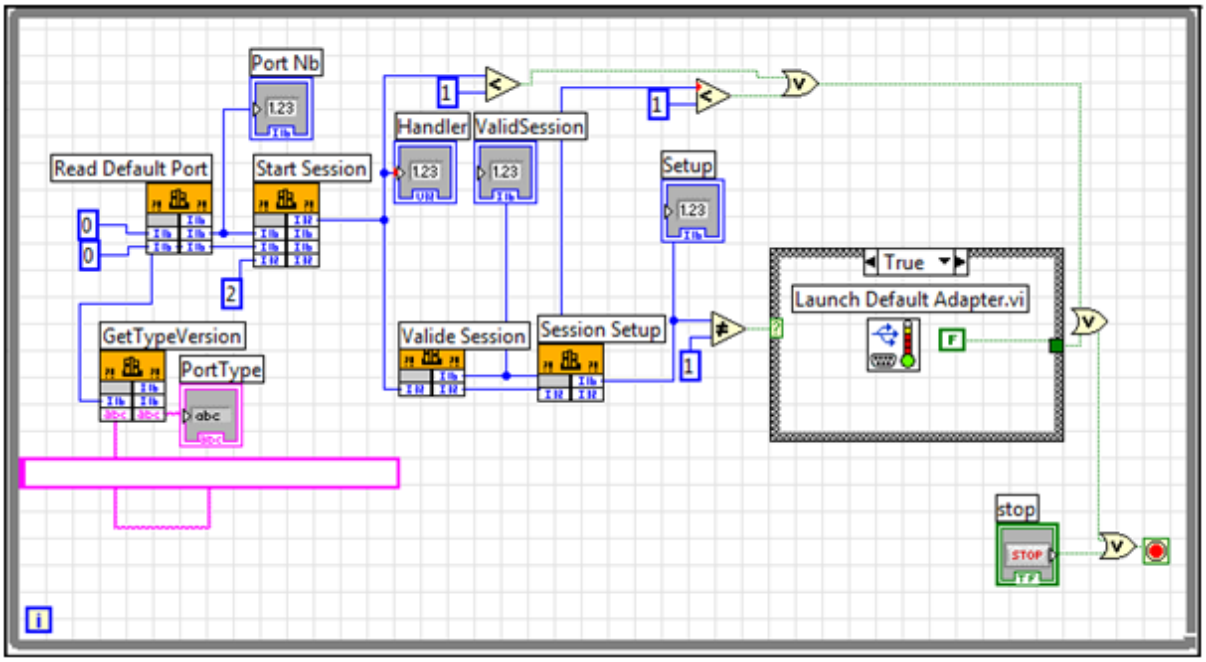
Na obrázku 21 můžeme vidět blokové schéma tohoto podprogramu. Celé je uzavřeno v cyklu *while*. Schéma obsahuje několik bloků *call library function*. Tyto bloky vyvolávají funkce z knihovny ibfs32.dll.

První blok vyvolává funkci *TMReadDefaultPort*, která čte defaultní číslo a typ portu ze systémového registru. Obě tyto informace jsou představovány čísly a popisují, na kterém portu je 1-wire síť připojena a jaký adaptér je používán. Z tohoto bloku vystupují další dva bloky. První je *Get Type Version*, který volá funkci *TMGetTypeVersion* a ta zobrazí textový

řetězec popisující typ připojeného portu a převodníku. Druhý je pak *Start Session* volající funkci *TMExtendedStartSession*. Tato funkce má jako argument číslo a typ portu a vrací číselnou hodnotu, pokud byla zahájena činnost anebo 0, označující že na 1-wire síti již probíhá další relace. Pokud je hodnota menší než nula, znamená to, že neexistuje žádný ovladač pro daný port.

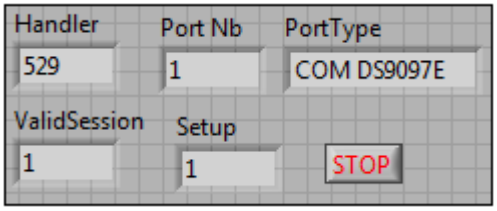
Další je blok *Valide Session*, volající funkci *TMValidSession*. Tato funkce určuje, zda je úloha stále platná na základě výstupu z bloku *Start Session*. Pokud je úloha platná, tak vrací 1, jinak 0. Další je blok volající funkci *TMSetup*, která ověřuje fyzickou integritu 1-wire sítě.

Jednotlivé výstupy z uvedených funkcí jsou porovnávány pomocí logických funkcí a v dalším kroku je takto řízena *Case* struktura, ve které je volán podprogram *Launch Default Adapter.vi*. [15]



Obrázek 21: Blokový diagram podprogramu *StartSession.vi*

Čelní panel tohoto podprogramu je zobrazen na obrázku 22 a obsahuje informace o výstupu z jednotlivých bloků a také o typu portu.

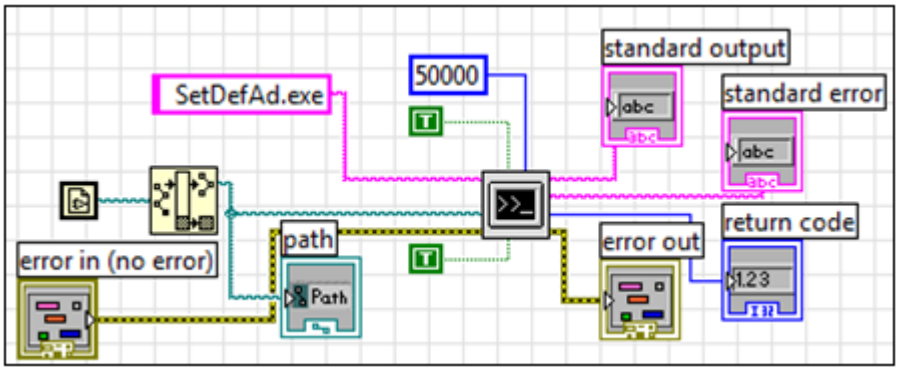


Obrázek 22: Čelní panel podprogramu *StartSession.vi*

Podprogram *Launch default adapter.vi* bude popsán dále.

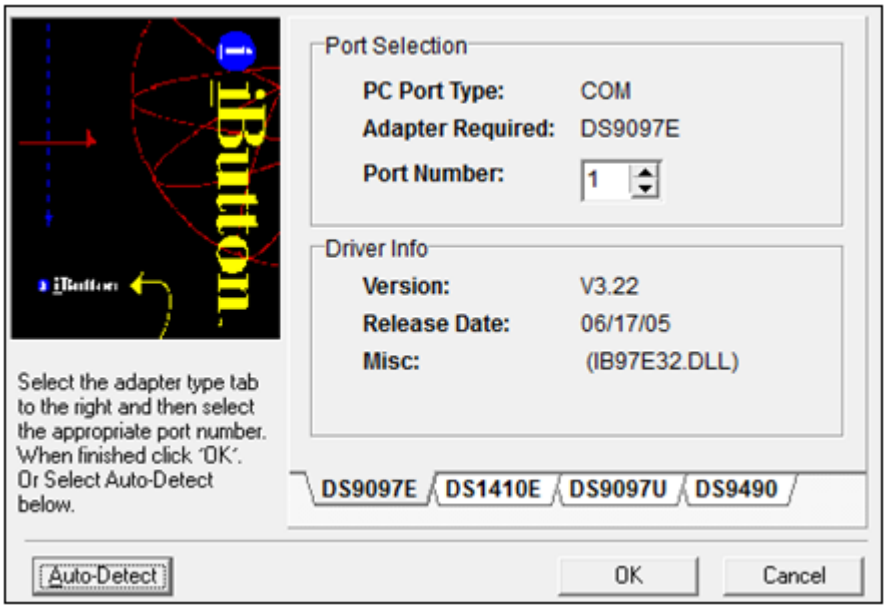
11.2.2 Popis podprogramu *Launch default adapter.vi*

Tento program navazuje na podprogram *StartSession.vi*. Jeho hlavním prvkem, jak můžeme vidět na obrázku 23, je blok *System Exec*. Hlavní funkcí tohoto bloku je vyvolání a spouštění programů nebo funkcí umístěného ve Windows. Další bloky slouží k samotnému ovládání vlastního *System Exec*. Patří sem název spouštěného programu a potom podmínky jeho spouštění. Mezi podmínky patří, aby program běžel minimalizovaný a čekalo se na jeho kompletizaci. Další bloky slouží k zobrazení výstupů.



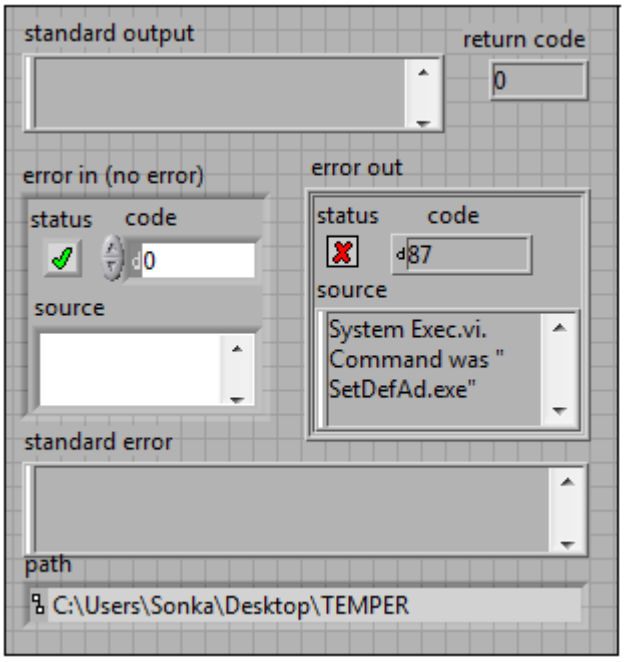
Obrázek 23: Blokové schéma podprogramu *Launch default adapter.vi*

Program spouštěný pomocí bloku *System Exec* je zobrazen na obrázku 24. Jedná se o program vytvořený firmou Dallas Instruments pro ovládání zařízení na 1-wire sběrnici. Obsahuje ovladače pro různé převodníky. Pro funkci našeho programu je důležitý převodník DS9097E, který je navázán na knihovnu *IB97E32.dll*. Tento program automaticky rozpozná jaký typ převodníku je připojen k počítači.



Obrázek 24: Program SetDefAd.exe

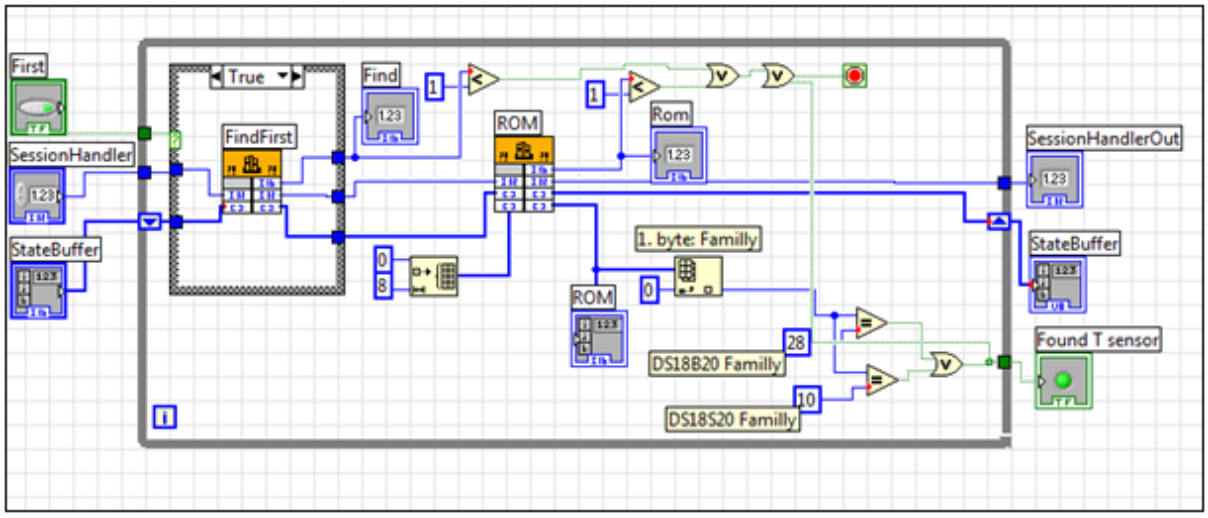
Čelní panel podprogramu *Launch default adapter.vi* indikuje program, který byl spuštěn a zobrazuje cestu ke složce, ve které se vyvolaný program nachází. Obrázek 25 zobrazuje čelní panel tohoto podprogramu.



Obrázek 25: Čelní panel podprogramu *Launch default adapter.vi*

11.2.3 Popis podprogramu *Find Temperature probe.vi*

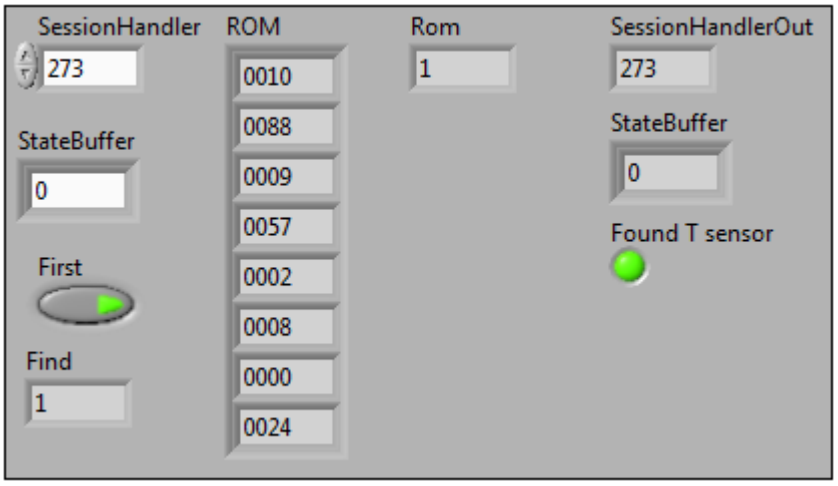
Tento podprogram je další, na který se odvolává hlavní program *Temper.vi*. Nejprve bude popsáno blokové schéma tohoto podprogramu zobrazené na obrázku níže.



Obrázek 26: Blokové schéma podprogramu *Find Temperature Probe.vi*

Vstupem jsou hodnoty *SessionHandler* z podprogramu StartSession a *StateBuffer*. Další je logický datový typ *boolean*, který určuje programu, zda jde o první zařízení na 1-wire sběrnici. To program zjistí tak, že porovná index aktuálního cyklu. Pokud je tento index roven nule, jedná se právě o první zařízení a *case* struktura v programu je nastavena na hodnotu *true*. V ostatních případech, v našem případě pro zbylých patnáct čidel je hodnota *case* struktury nastavena na *false*. V prvním případě je v *case* struktuře umístěn blok *call library function*, který volá funkci *TMFirst*. Tato funkce najde první zařízení na 1-wire sběrnici specifikované hodnotou *SessionHandler*. Výstupem funkce je jednička, pokud bylo nalezeno zařízení a nula pokud zařízení na síti nebylo nalezeno. V ostatních případech volá blok funkci *TMNext*, která má obdobnou funkci jako *TMFirst*. Když je adresováno poslední zařízení, vrátí blok hodnotu nula a celý cyklus pro šestnáct čidel se zopakuje a začne opět voláním funkce *TMFirst*.

Další blok *call library function* volá funkci *TMRom*. Tato funkce přenáší data z vnitřního osmibajtového bufferu zařízení do pole ROM, které je reprezentováno osmi vybranými čísly zobrazenými v programu. Směr přenosu je řízen pomocí hodnoty prvního čísla v řetězci z bufferu. Jedná-li se o nulu, je osm bajtů z vnitřního bufferu převedeno do pole osmi celočíselných proměnných. Pokud je hodnota nenulová, pak je osm bajtů převedeno do vnitřního osmibajtového bufferu. Tato funkce umožňuje programu získat data ze zařízení, která byla nalezena funkcí *TMFirst* a *TMNext*. Rovněž to umožňuje programu specifikovat ROM data konkrétního zařízení na sběrnici a adresovat toto konkrétní zařízení. Jako další krok je vybrán první bajt tzv. *familly byte*, podle jeho hodnoty se rozhoduje, zda jsou připojena čidla DS18S20 nebo DS18B20. Pokud je tato hodnota 10 jedná se o námi použitá čidla DS18S20, pro čidla DS18B20 je hodnota 28. Dále je v tomto programu vloženo několik podmínek, podle kterých je rozhodováno a indikováno, zda jsou čidla přítomna na sběrnici. Indikace je znázorněna pomocí LED diody na čelním panelu. Dalším výstupem je opět *SessionHandler* a *StateBuffer*. Program je potom uzavřen v cyklu *while*. [15]

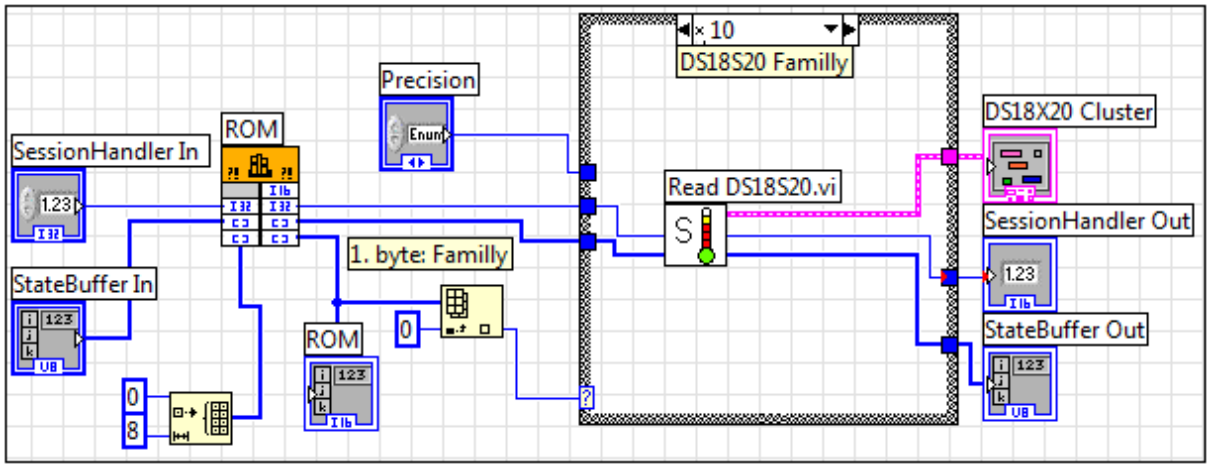


Obrázek 27: Čelní panel programu *Find temperature probe.vi*

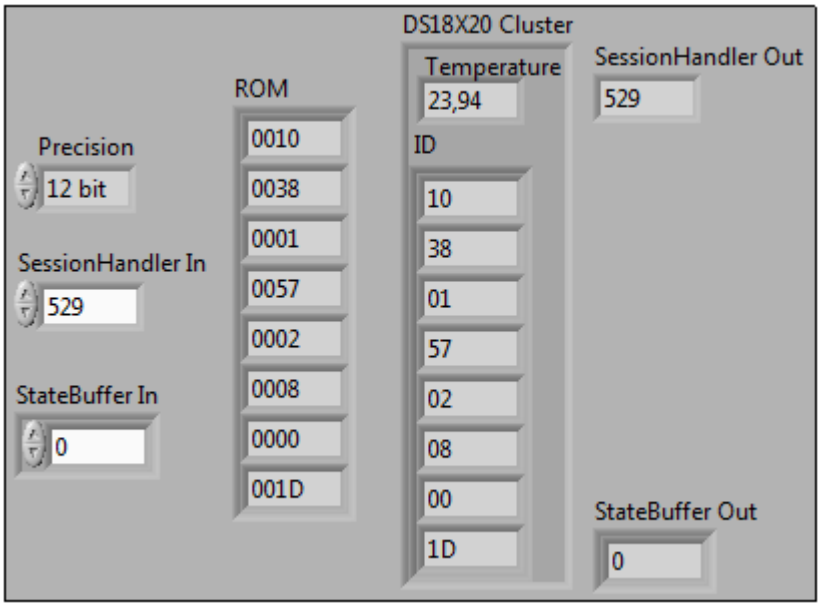
Obrázek 27 znázorňuje čelní panel podprogramu. V tomto případě jsou zde zobrazeny hodnoty pro první čidlo na sběrnici. To je indikováno zeleně podsvíceným přepínačem vlevo. Dále je zobrazeno pole ROM, kde jsou vypsány hodnoty všech osmi bajtů. První bajt odshora představuje *familly* bajt. V levé části potom LED dioda indikuje, že bylo nalezeno zařízení na sběrnici. Pro další čidla se změní stav přepínače *First* a změní se některé bajty pole ROM.

11.2.4 Podprogram *Read temperature.vi*

Tento podprogram umožňuje zobrazit a zpracovat hodnoty získané pomocí jeho dvou podprogramů *Read DS18S20* a *Read DS18B20*. Podprogram si můžeme prohlédnout na obrázku 28. Tento podprogram umožňuje vybírat mezi čidly DS18S20 a DS18B20. Jeho základem je *case* struktura, která je řízena opět pomocí *familly* bajtu. Vstupem je *SessionHandler* a *StateBuffer*.



Obrázek 28: Blokový diagram programu *Read temperature.vi*

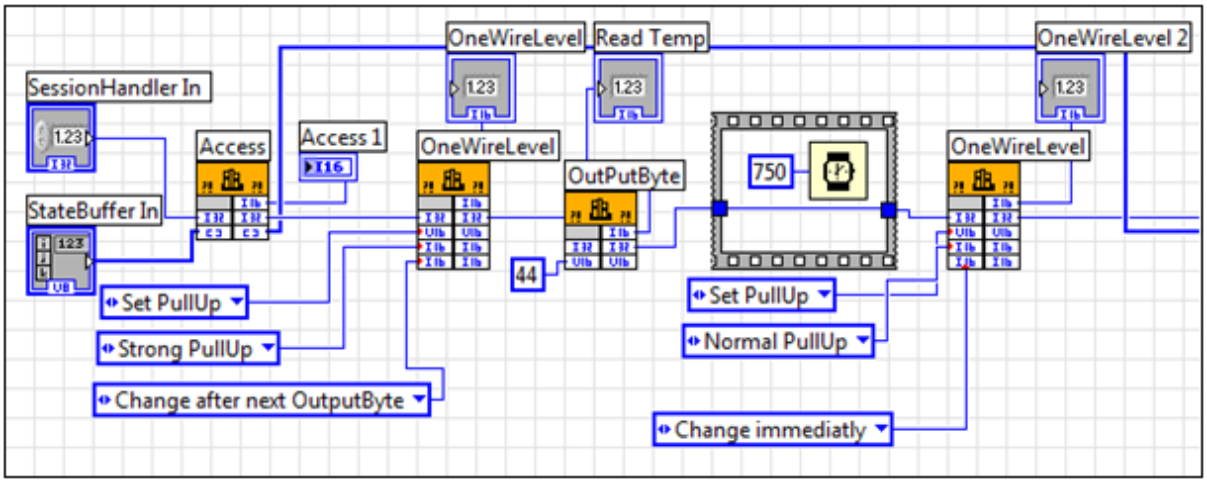


Obrázek 29: Čelní panel programu *Read temperature.vi*

Čelní panel tohoto podprogramu je zobrazen na obrázku 29. Na čelním panelu nalezneme pole *ROM* a také pole označené *DS18X20 Cluster*. Toto pole je vybíráno z podprogramů a zobrazuje údaj o teplotě aktuálně měřeného čidla a jeho ID. Dále je tu prvek *numeric control*, jímž lze volit přesnost pro čidla DS18B20.

11.2.5 Podprogram *Read DS18S20.vi*

Tento podprogram je volán programem *Read temperature* a umožňuje číst teploty z čidel a zobrazovat je v programu. Jeho blokový diagram je značně rozsáhlý a proto jej zobrazíme na několik částí. První část je na obrázku 30.



Obrázek 30: *Read DS18S20.vi* blokový diagram – část 1.

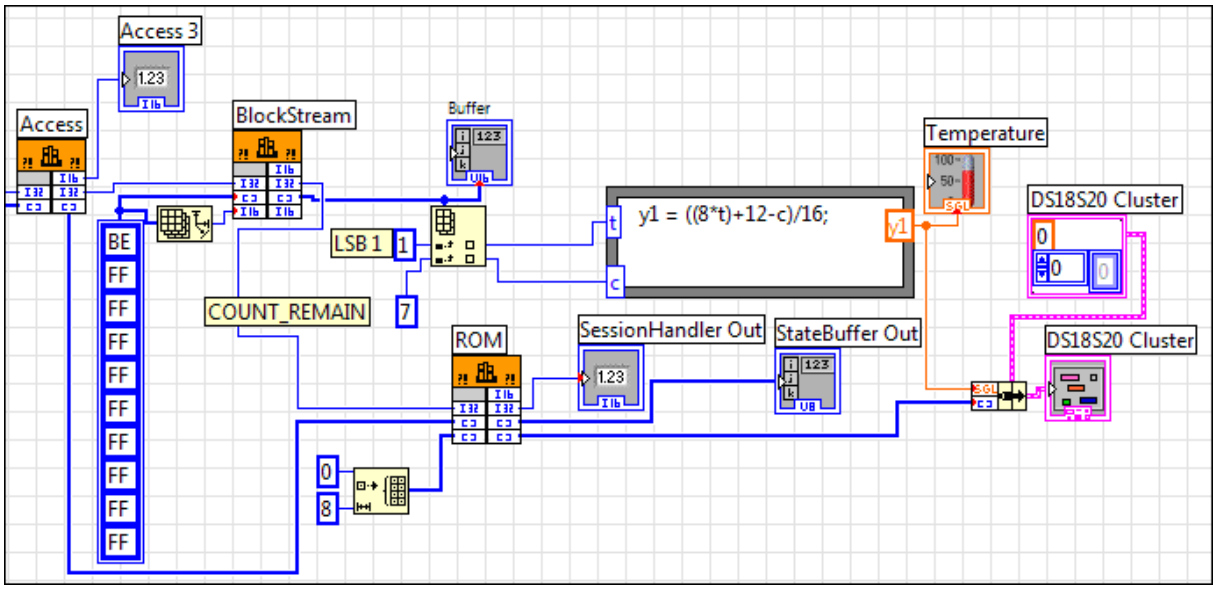
Vstupem jsou opět *SessionHandler* a *StateBuffer*. Hodnoty z nich vedou do bloku *Call library function*. Tento blok volá funkci *TMAccess* z knihovny *ibfs32.dll*. Fukce *TMAccess* vysílá TMTouchReset signál a vrací 1, pokud detekuje presence puls na 1-wire síti. Potom funkce zpřístupňuje zařízení, jehož ROM kód je ve vnitřním osmibajtové buffer paměti. Funkce umožňuje použití funkčních příkazů jako je např. *read scratchpad*. [15]

Další blok volá funkci *TMOneWireLevel*, která ovládá úroveň 1-wire sběrnice. Zde se nataví vysoká úroveň sběrnice ihned po dalším příkazu. Následuje blok volající funkci *TMTouchByte* jež umožňuje vyslat příkaz *Read temperature*. Dalši je *Flat sequence structure*, ve které je blok *Wait* umožňující pozdržet program na 750 ms.

Poté opět následuje blok *Call library function* volající funkci *TMOneWireLevel*, kdy se vrátí úroveň na 1-wire sběrnici.

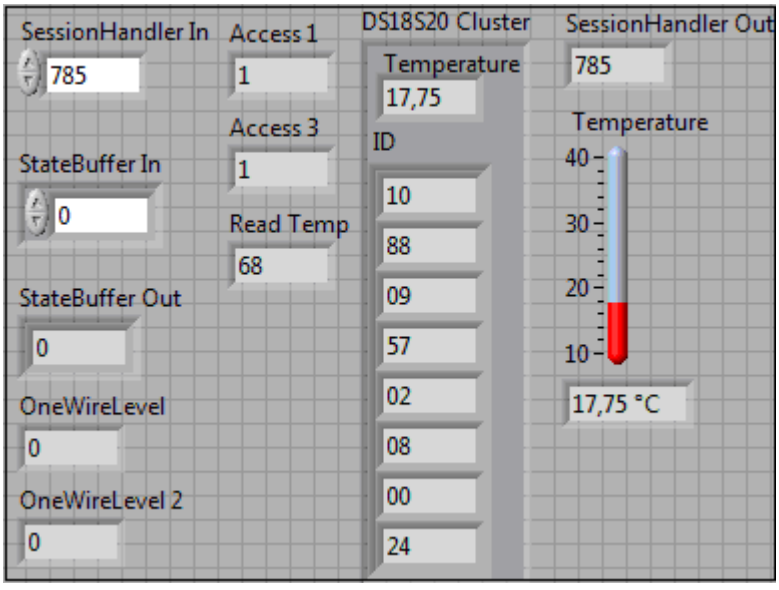
Popis pokračuje na obrázku 31. Blok označený *Access* zpřístupňuje zařízení na sběrnici. Následuje blok volající funkci *TMBlockStream*, která umožňuje číst *scratchpad* registr. Funkce převádí řetězec bajtů po teplotní konversi tak, aby mohly být vyčteny a zobrazeny. Následující blok indexuje buffer a umožňuje použít jen konkrétní bajty. Pro další použití a výpočet teploty jsou potřebné bajty *temperature_LSB* a bajt *count_REMAIN*. Hodnoty těchto bajtů vstupují do struktury *Formula Node*, která umožňuje zadat jednoduchou

rovnici a zobrazit vypočtené hodnoty. Postup výpočtu teploty je podrobně popsán v kapitole 11.2.6. Hodnoty jsou zobrazeny pomocí bloku *Temperature*, a dále postupují do bloku *Boundle*, který je zobrazí spolu s jejich ID na čelním panelu. [15]



Obrázek 31: *Read DS18S20.vi* blokový diagram - část 2.

Čelní panel tohoto podprogramu obsahuje indikátory vstupních proměnných a také pole obsahující údaj s naměřenou teplotou a řetězec ID aktuálního čidla. Dále je zde sloupcový indikátor zobrazující teplotu.



Obrázek 32: Čelní panel programu *Read DS18S20.vi*

Program *Read DS18B20* funguje obdobně jako program *Read DS18S20* s vyjímkou části pro výpočet teploty. Protože ale jsou všechna použitá čidla typu DS18S20 není nutné popisovat výpočet teploty pro čidla DS18B20.

11.2.6 Popis výpočtu teploty pro čidla DS18S20

Čidla DS18S20 mají základně nastavenou přesnost zobrazení teploty na 0.5 °C. Pro měření stability je tato přesnost nedostačující, proto je nutné dosáhnout vyššího teplotního rozlišení. Postup jak toho dosáhnout je popsán v kapitole 7.2.2, konkrétně pomocí vzorce číslo (7). Hodnota *TEMP_READ* se vypočítá tak, že se vydělí dvěma. Tak se odstraní rozlišení 0,5°C. Hodnoty *COUNT_PER_C* a *COUNT_REMAIN* slouží pro vypočítání vyššího teplotního rozlišení. Pro čidla DS18S20 je hodnota *COUNT_PER_C* vždy rovna 16. To nám umožňuje zjednodušit rovnici (7) do tvaru:

$$TEMPERATURE = TEMP\_READ - 0{,}25 + \frac{16 - COUN\_REMAIN}{16} \tag{8}$$

Dále celou tuto rovnici vynásobíme 16 a dostaneme tvar:

$$16 * TEMPERATURE = 16 * TEMP\_READ - 16 * 0{,}25 + 16 * \left(\frac{16 - COUN\_REMAIN}{16} \right) \tag{9}$$

Takto upravenou rovnici zjednodušíme na:

$$16 * TEMPERATURE = 16 * TEMP\_READ + 12 - COUNT\_REMAIN \tag{10}$$

Nesmíme zapomenout, že v programu je *TEMP_READ* počítána jako obsah LSB, který se dělí dvěma. Proto dostáváme tvar:

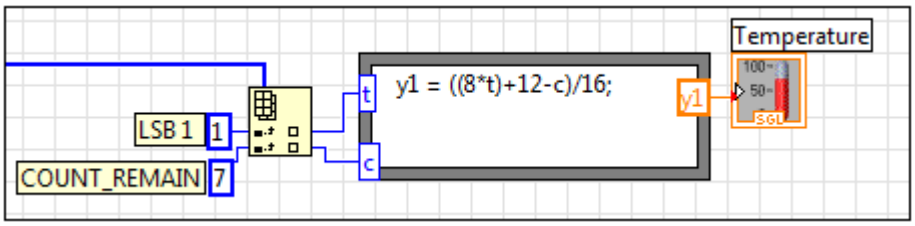
$$16 * TEMPERATURE = 16 * \frac{TEMP\_READ}{2} + 12 - COUNT\_REMAIN \tag{11}$$

Tuto rovnici musíme ještě upravit a dostaneme tvar:

$$TEMPERATURE = \frac{8 * TEMP\_READ + 12 - COUNT\_REMAIN}{16} \tag{12}$$

Bloky, které to umožňují, jsou zobrazeny na obrázku 33. Pomocí bloku *index array* jsou vybrány konkrétní hodnoty z bufferu, které jsou potřebné pro výpočet. Hodnoty LSB a

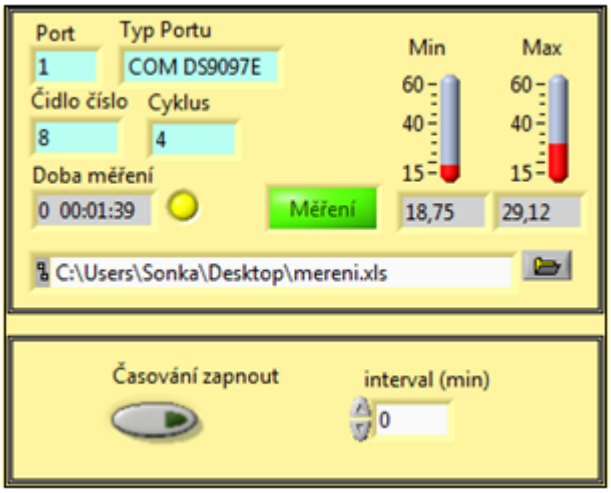
COUNT_REMAIN jsou přivedeny na vstup bloku *formula node*, který provede výpočet dle rovnice 12. Výsledek je zobrazen na čelním panelu.



Obrázek 33: Bloky pro výpočet teploty čidla DS18S20

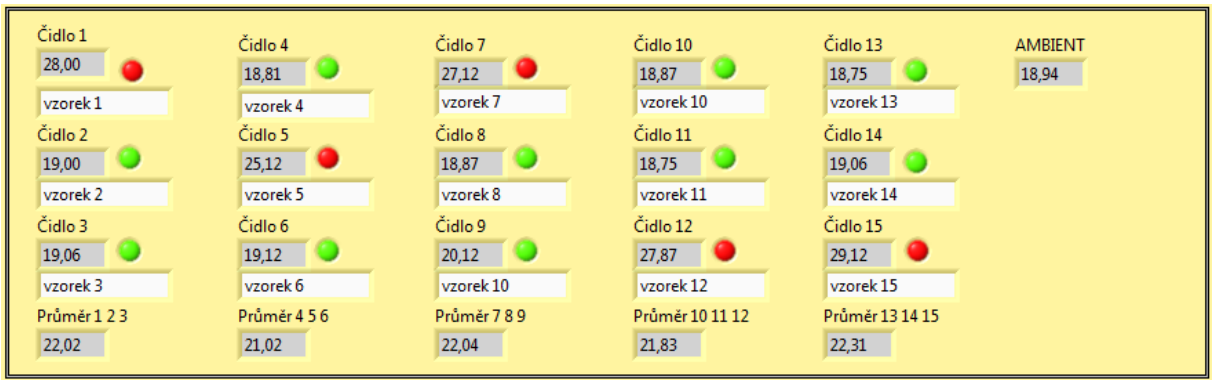
11.3 Popis hlavního programu *Temper.vi*

Hlavní program ovládá ostatní podprogramy a umožňuje zobrazit naměřené výsledky. Tyto jsou přehledně znázorněny v grafech a tabulce. Dále můžeme pomocí prvků na jeho čelním panelu zvolit xls soubor, do kterého budou výsledky měření ukládány a naformátovat hlavičku této tabulky. Vzhled základní části čelního panelu je vidět na obrázku 34. Vlevo nahoře je zobrazen převodník a číslo portu, na kterém je připojen. Dále je zde indikováno číslo čidla, jež je aktuálně měřeno a je zobrazeno kolikátý cyklus měření probíhá. Další políčko zobrazuje, jak dlouho celé měření probíhá. Pokud je aktivní záznam, bliká dioda vedle políčka *Doba měření* a svítí obdélníková ikona *Měření*. Kdyby se vyskytla chyba v programu, obdélníkový indikátor *Měření* zhasne, i když bude pokračovat odpočítávání času v kolonce *Doba měření*. Dále jsou zobrazeny maximální a minimální teplota aktuálního měření. Pod těmito indikátory najdeme políčko, pomocí něhož zvolíme umístění a název výstupního souboru xls. Další je oblast pro nastavení časových prodlev jednotlivých měření. V pravém poli nastavíme časový interval v minutách a zapneme tlačítko vlevo. Toto se projeví rozsvícením zeleného indikátoru na přepínači.



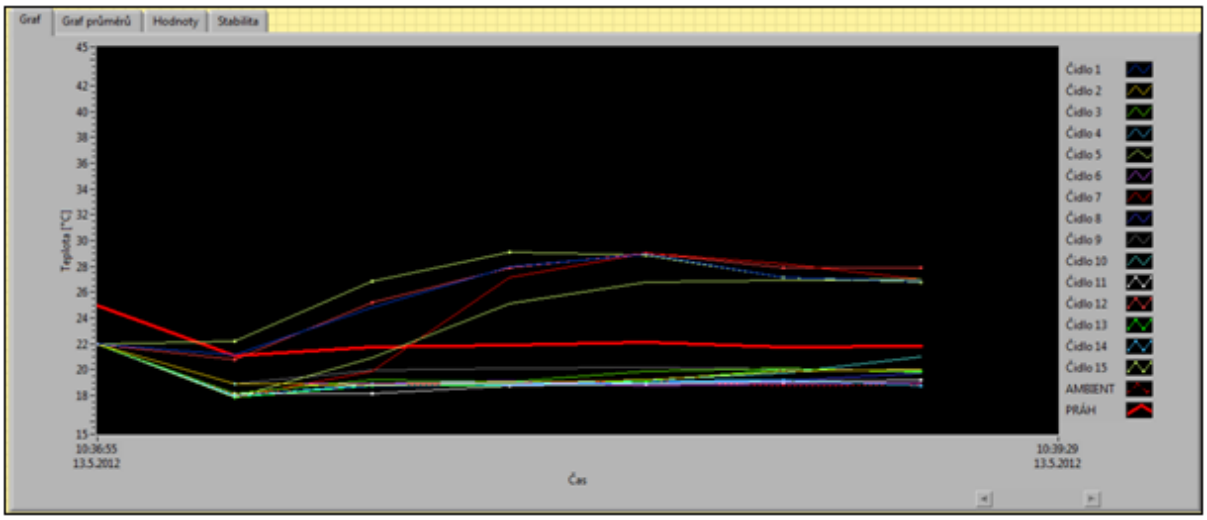
Obrázek 34: Čelní panel programu *Temper.vi* - informativní část

Obrázek 34 popisuje další část čelního panelu programu *Temper.vi*. Zde jsou zobrazeny aktuální hodnoty všech šestnácti čidel. Poslední čidlo, kterým se měří teplota okolního prostředí je pojmenováno *AMBIENT*. Rozmístění indikátorů odpovídá reálnému rozestavení kontejnerů se vzorky, do kterých se umísťují čidla. Lze tak předejít případnému špatnému umístění vpichové sondy do nesprávného kontejneru. Dále je u každého rámečku s aktuální teplotou *LED indikátor*, který zobrazuje stav stability. Pokud vzorek nepřekročil práh stability určený čidlem AMBIENT, zůstává indikátor zelený. Při překročení začne indikátor svítit červeně. Pod čidly jsou umístěny rámečky, pomocí kterých lze formátovat hlavičku výstupní tabulky. Lze tak jednoduše pojmenovat jednotlivé vzorky. Spodní řádek tvoří indikátory zobrazující průměrné hodnoty z jednotlivých čidel.



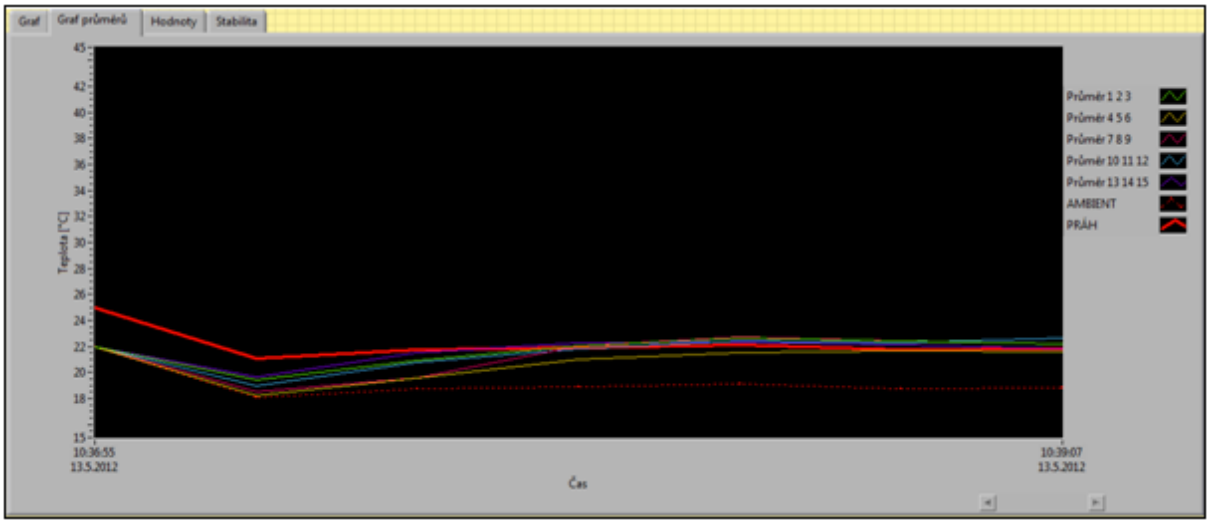
Obrázek 35: Čelní panel programu *Temper.vi* – zobrazení teplot čidel

Další prvek na čelním panelu hlavního programu je prvek *Tab control* zobrazený na obrázku 36. Tento prvek obsahuje celkem čtyři záložky, mezi kterými lze přepínat. Zleva jsou to záložky Graf, Graf průměrů, Hodnoty a Stabilita. Na obrázku 36 je zobrazená záložka Graf zobrazující vývoj teploty ze všech šestnácti čidel v čase. Legenda pro čidla je zobrazená vpravo. V grafu je navíc silnou červenou čarou zobrazen práh. Tato hodnota je v každém bodě měření vypočtena pomocí ambientního čidla. Dole je potom posuvník pro prohlížení vývoje grafu.



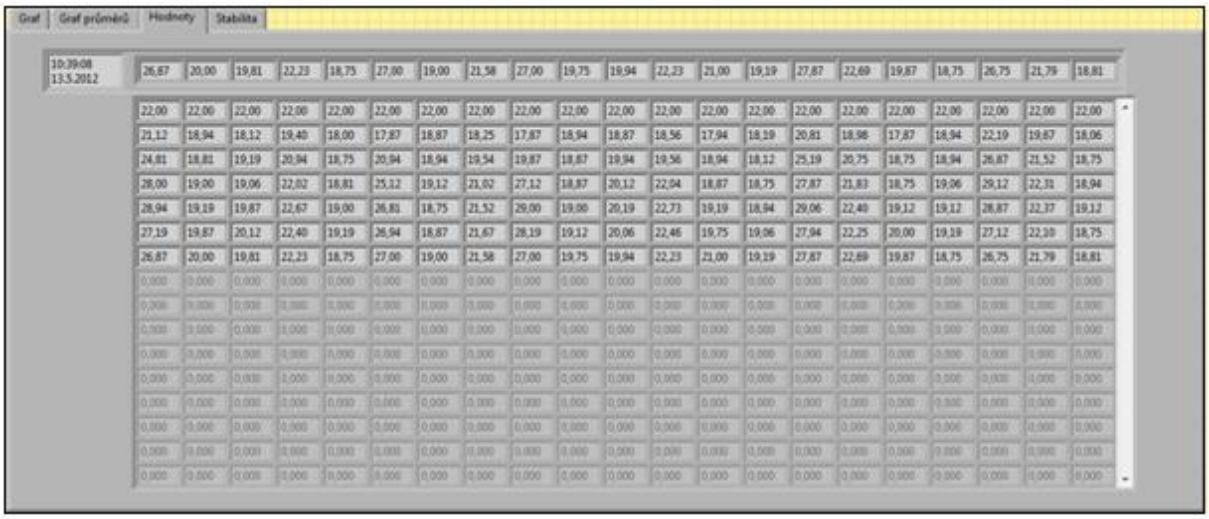
Obrázek 36: Čelní panel programu *Temper.vi* – graf

Obrázek 37 níže zobrazuje pouze graf průměrů ze skupin čidel. Opět je zde legenda a práh zobrazen silnou červenou linkou.



Obrázek 37: Čelní panel programu *Temper.vi* – graf průměrů

Záložka znázorněná na obrázku 38 zobrazuje tabulku záznamů z čidel. Vlevo je pole zobrazující čas a datum aktuálního měření a hodnoty z čidel v řádku. Hodnoty se potom zapisují do tabulky odshora, takže aktuální hodnota je vždy na spodním řádku této tabulky. Pomocí roletky vpravo lze tabulku jednoduše prohlížet.



Obrázek 38: Čelní panel programu *Temper.vi* – Hodnoty

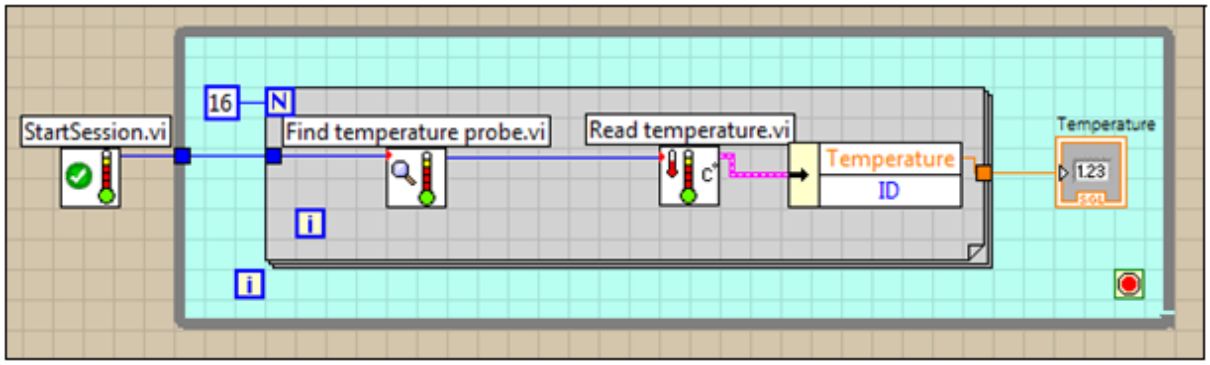
Poslední záložka na obrázku 39 níže představuje překročení aerobní stability jednotlivých čidel. Pole vlevo nahoře zobrazuje začátek měření. Pro každé čidlo zvlášť je potom zobrazen čas kdy čidlo zaznamenalo překročení stability vzorku. Překročení stability je indikováno červenou LED diodou a doba stability je zobrazena v políčku v hodinách, minutách a sekundách.

Obrázek 39: Čelní panel programu *Temper.vi* – Stabilita

11.4 Popis blokového diagramu programu *Temper.vi*

Celý blokový diagram lze nejlépe popsat pomocí obrázku 40. Na začátku programu je podprogram *StartSession*. Hodnoty z něj vedou do cyklu *while*, který zaručuje kontinuální měření po dobu spuštění programu. Uvnitř tohoto cyklu je další cyklus tentokrát *for*, který obsahuje podprogramy *Find temperature probe* a *Read temperature*. Podprogram Read temperature obsahuje dva vnořené podprogramy *Read DS18S20* a *Read DS18B20*. Tento

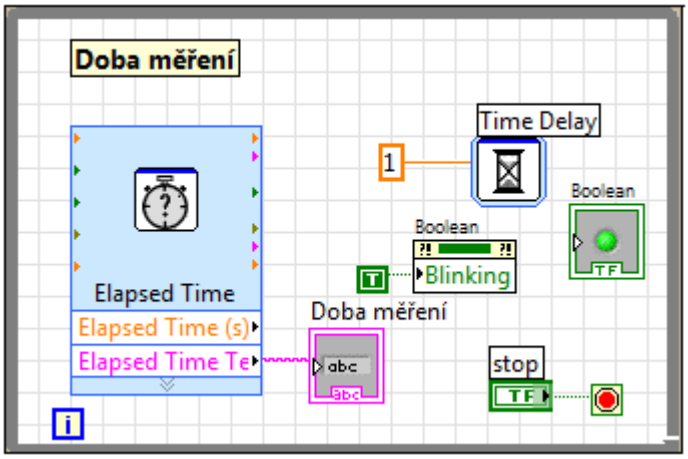
cyklus šestnáctkrát provede operace pro adresování čidel a výpočet jejich teplot. Hodnoty vystupují z tohoto cyklu a dále jsou zpracovány a zobrazeny na čelním panelu. Program vždy při spuštění v první cyklu měření zobrazuje jednu vypočtenou hodnotu do indikátorů všech čidel. Toto je způsobeno nastavováním hodnot v podprogramu. V blokovém diagramu je toto nastavování nahrazeno blokem, který při prvním měření nastaví do všech čidel vždy hodnotu 22. Čidla tak mají jeden interval měření, aby se ustálila jejich teplota. Popis jednotlivých bloků a struktur nalezneme níže.



Obrázek 40: Přehled blokového diagramu programu *Temper.vi*

11.4.1 Popis bloku Doba měření

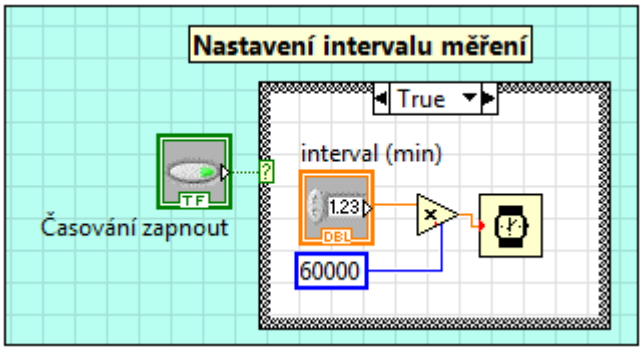
Tento blok můžeme vidět na obrázku 41 a jeho úkolem je zobrazit na čelním panelu údaj o uplynulé době měření. Základem je blok *Time Delay*, který umožňuje do programu vložit časovou prodlevu. Prodleva je nastavena na jednu sekundu. Celý blok je uzavřen v cyklu *while*, a tak se prodleva opakuje. Tím definujeme odpočítávání po jedné sekundě. Další blok *Elapsed Time* umožňuje měření uplynulé doby běhu této smyčky. Hodnota je dále zobrazena na čelním panelu. Dalším prvkem ve smyčce je *Boolean* indikátor, který je programově nataven na blikání. Tím docílíme indikace běhu celého programu na čelním panelu, viz obrázek 33.



Obrázek 41: Blok *Doba měření*

11.4.2 Blok pro nastavení intervalu měření

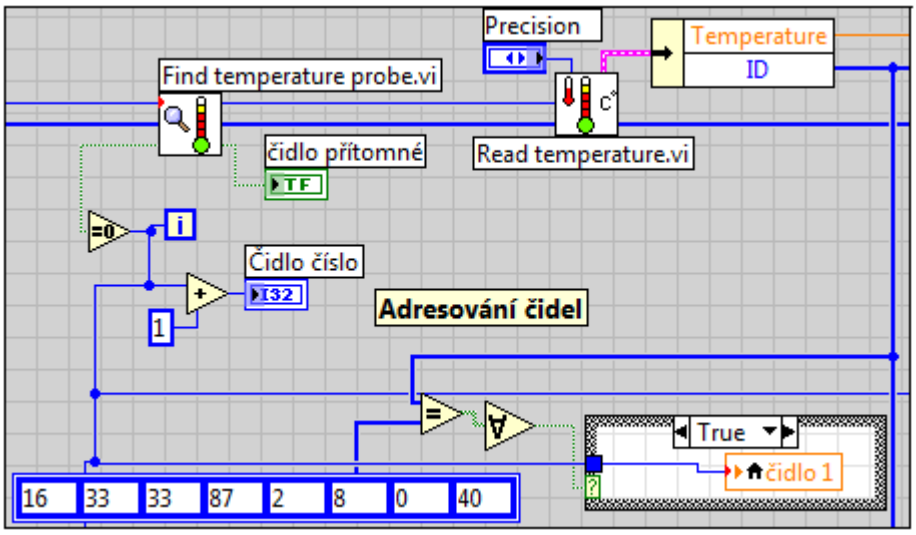
Tento blok umožňuje programově nastavit interval mezi jednotlivými záznamy. Jeho základ tvoří blok *Wait*, který čeká specifický počet milisekund. Proto je nastavený interval násoben 60000, aby na čelním panelu bylo možné zvolit prodlevu v minutách. Tyto bloky jsou uzavřeny ve struktuře *case*, která je ovládána z čelního panelu. Pokud je ovládací prvek sepnut, pak po uplynutí zvoleného časového intervalu, proběhne celý cyklus měření znovu. Pokud není, pak měření probíhá kontinuálně. Vzhled ovládacího prvku *case* struktury najdeme na obrázku 33 dole.



Obrázek 42: Blok pro nastavení intervalu měření

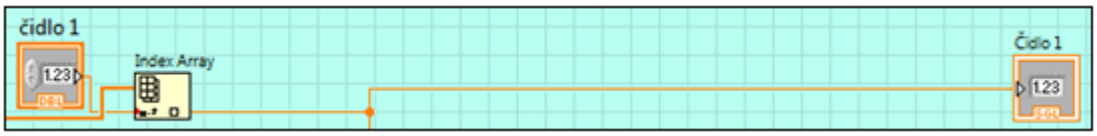
11.4.3 Blok pro adresování čidel v programu

Naměřené hodnoty se podle požadavků firmy Nutrivet s.r.o. musí pro jednotlivá čidla objevovat vždy ve stejném indikátoru ve stejném místě na čelním panelu programu. Aby bylo toto možné, bylo nutné čidla adresovat pomocí jejich specifických ID. To zajišťuje několik bloků uvnitř *for* cyklu programu *Temper.vi*. Specifické ID pro každé čidlo bylo vloženo do 1D pole konstant. Toto pole je potom porovnáváno s proměnnými vycházejícími z podprogramu *Read temperature*, ze kterého postupně vycházejí ID čísla všech čidel připojených na 1-wire sběrnici. Pokud je vybírána hodnota z čidla se stejným ID jako je v našem konstantním poli, blok *And array elements*, který ovládá přiloženou *case* strukturu, nastaví na hodnotu *true*. Blok *And array elements* nastaví *true* pouze pokud jsou obě hodnoty do něj vstupující *true*. V případě, že jsou splněny podmínky a *case* struktura je nastavena na hodnotu *true*, je do lokální proměnné zaznamenán index čidla, jehož hodnoty jsou zobrazovány.



Obrázek 43: Blok adresování čidel programu *Temper.vi*

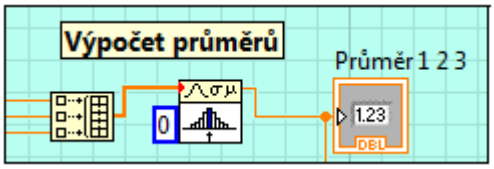
Tato lokální proměnná potom řídí blok *Index Aray* a funguje zde právě jako index daného čidla, viz obrázek 44. Hodnoty z bloku *Index Array* jsou potom zobrazeny na čelním panelu a je s nimi dále pracováno. Takto jsou v programu indexována všechna použitá čidla.



Obrázek 44: Indexování pomocí bloku *Index Array*

11.4.4 Výpočet průměrů ze skupin čidel

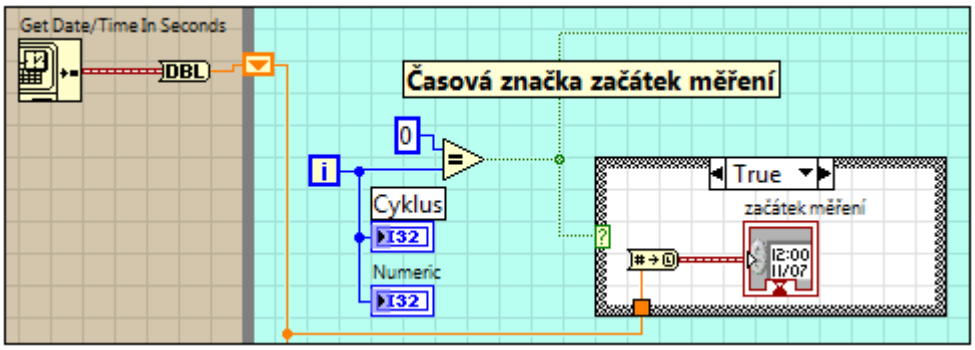
Podle požadavků má být program schopen počítat průměr hodnot vždy z trojice čidel a tuto hodnotu dále zobrazit na čelním panelu. Výpočet průměru můžeme vidět na obrázku 45. Na vstupu jsou hodnoty z čidel, v tomto případě z čidel číslo 1, 2 a 3. Dále je zde blok *Build Array*, který spojí hodnoty do jednoho pole a výstup z něj vede do bloku *Measures of mean*, který počítá průměr. Zde je nastavena numerická konstanta nula, která značí, že je počítán aritmetický průměr. Hodnota průměru je zobrazena na čelním panelu a je s ní dále počítáno v jiných oblastech programu.



Obrázek 45: Blok pro výpočet průměrů z čidel

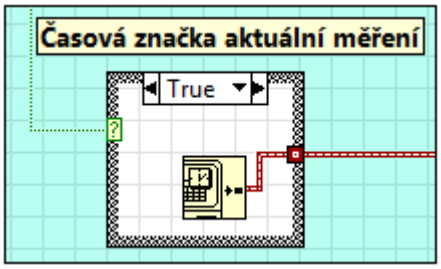
11.4.5 Blok pro výpočet časových značek

Pro správnou funkci a zobrazování hodnot v programu je nutné si vytvořit několik časových značek. První časová značka určuje čas, kdy začalo měření. Její výpočet je zobrazen na obrázku 46. Na vstupu je blok *Get Date/Time In Seconds*, který vypočítává počet sekund, který uplynul od 1.1.1904 12:00 hodin, což je tzv. universální čas programu LabVIEW. Tato hodnota je dále pomocí bloku *To Double Precision Float* převedena a vstupuje do cyklu *while* a dále do *case* struktury, kde je převedena zpět na časovou značku. *Case* struktura je zde opět řízena a nastaví hodnotu *true* jen v případě, že se jedná o první cyklus smyčky *while*. To je zajištěno porovnáním iteračního terminálu s konstantou nula.



Obrázek 46: Blok pro získání časové značky – začátek měření

Další je časová značka pro aktuální měření. Ta je řízena *boolean* indikátorem vycházejícím z *for* cyklu uvnitř programu *Temper.vi*. Pokaždé, když je dokončen tento cyklus, tak se hodnota *true* aktualizuje, a ukládá tak aktuální časovou značku. Hodnoty časové značky přicházejí z posuvného registru, který umožňuje uchovávat v paměti posloupnost časových značek.

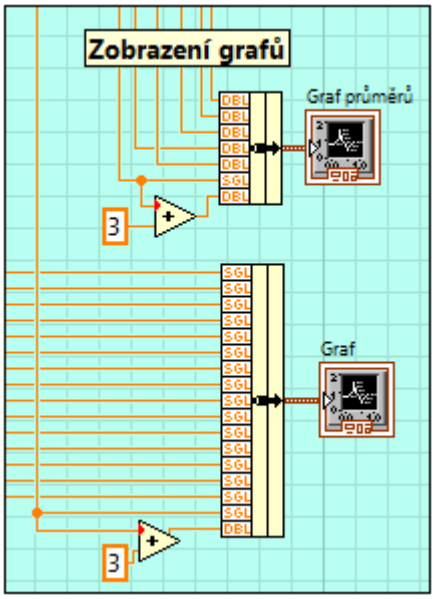


Obrázek 47: Blok pro získání časové značky – aktuální měření

11.4.6 Blok pro zobrazení grafů

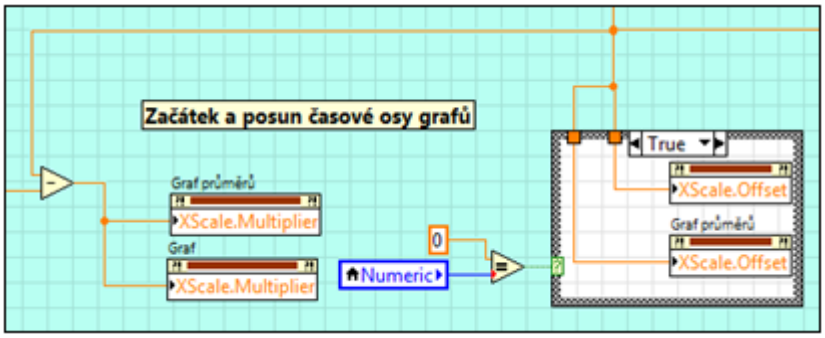
Další zadání bylo, aby byl program schopen zobrazovat vývoj teplot v čase. Bloky zajišťující tuto funkci nalezneme na obrázku 48. Hodnoty z jednotlivých čidel jsou přiváděny do bloku *Bundle*, který seskupí těchto několik hodnot a výstup převede do bloku *Waveform chart* zobrazující graf na čelním panelu viz obrázek 36 a 37. Vrchní část obrázku 48 tvoří graf

průměrů ze skupin čidel a spodní vytváří graf ze všech šestnácti čidel. Oba grafy jsou doplněny o prahovou křivku. Její hodnota se vypočítává z hodnot ambientního čidla, jehož hodnota se zvětší pomocí bloku *Add* o 3.



Obrázek 48: Blok pro zobrazení grafů programu *Temper.vi*

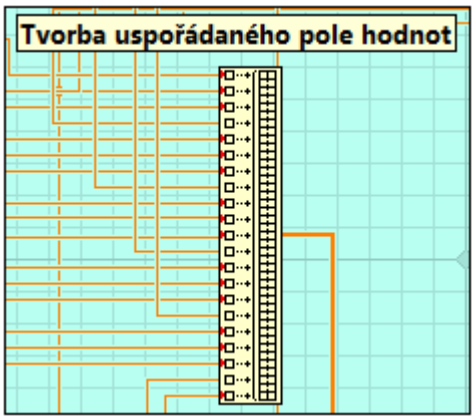
Aby se na časové ose obou grafů zobrazovaly správné údaje, byly bloky *Waveform chart*, doplněny o bloky *XScale.Offset* a *XScale.Multiplier*. Hodnota *XScale.Offset* určuje začátek časové osy v grafech a je vypočtena na základě hodnot z posuvného registru. Je umístěna v *case* struktuře, která je řízena stejně jako časová značka začátek měření. Takto se zajistí aby *XScale.Offset* byl v grafu konstantní. *XScale.Multiplier* získává hodnoty z posuvného registru. Tím že odečte aktuální počet sekund, které uplynuly od universálního času LabVIEW, od počtu sekund začátku měření získá posun pro každé jednotlivé měření. Poslední hodnoty na časové ose jsou tedy aktuální podle systémového času v době měření.



Obrázek 49: Bloky pro výpočet začátku a posunu časové osy

11.4.7 Blok pro tvorbu uspořádaného pole hodnot

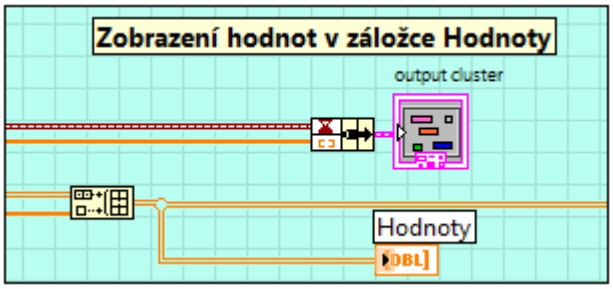
Aby bylo možné zapisovat naměřené hodnoty správně do polí a tabulek, musíme nejprve tyto naměřené hodnoty uspořádat. To se děje pomocí bloku *Build Array*. Na jeho vstupy jsou přivedeny indexované hodnoty z jednotlivých čidel, viz kapitola 11.4.3. Výstup potom tvoří řada hodnot v pořadí, ve kterém jsme zapojili vstupy. Toto pořadí zůstává dále stejné.



Obrázek 50: Blok pro tvorbu uspořádaného pole hodnot

11.4.8 Blok pro zobrazení tabulky v záložce Hodnoty

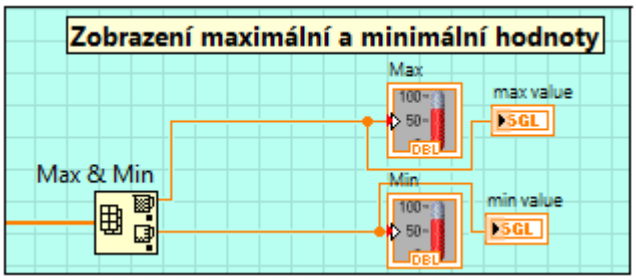
Na čelním panelu hlavního programu se nachází záložka hodnoty (obrázek 38). Pro správné zobrazení těchto hodnot jsou na blokovém diagramu bloky, které můžeme vidět na obrázku 51. Na vrchní části záložky hodnoty je řada zobrazující aktuální měření. Je zobrazen čas a datum měření a naměřené hodnoty. Toto je realizováno pomocí bloku *output cluster*. Jeho vstup tvoří blok *Bundle*, spojuje data z časové značky aktuálního měření a data z teplotních čidel, pocházející z bloku na obrázku 49. Pod tímto polem je tabulka, do níž se ukládají hodnoty jednotlivých měření. Aby se hodnoty zapisovaly pod sebe, jsou vybírány z posuvného registru.



Obrázek 51: Blok pro zobrazení hodnot v záložce Hodnoty

11.4.9 Bloky počítající maximální a minimální hodnoty

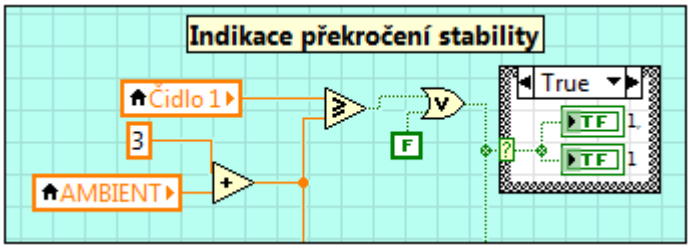
Na čelním panelu (viz obrázek 34) jsou zobrazeny maximální a minimální hodnoty aktuálního měření. Jsou vypočteny pomocí bloku *Array Max & Min*. Tento blok dokáže porovnat hodnoty z jeho vstupu a vybrat mezi nimi maximální a minimální. Hodnoty jsou potom graficky zobrazeny na čelním panelu.



Obrázek 52: Bloky pro výpočet maximální a minimálních teplot

11.4.10 Bloky pro indikování překročení aerobní stability

Na čelním panelu jsou na několika místech indikována překročení stability. To můžeme vidět na obrázku 35 a 39. Bloky umožňující indikování vidíme na obrázku 53. Na vstupu jsou naměřená data z čidel 1 až 15 a potom z ambientního čidla, vůči kterému počítáme stabilitu. K hodnotě ambientního čidla přičteme konstantu 3 a porovnáme pomocí bloku *Greater or equal*. Pokud má některý měřený vzorek teplotu o tři stupně celsia vyšší než je hodnota ambientního čidla, tento blok vysílá logické *true* a to vstupuje do bloku *Or*. Pokud jsou oba vstupy bloku *Or* typu *false*, je i jeho výstup typu *false*. V ostatních případech je výstup *true* a ten řídí *case* strukturu, ve které jsou vloženy *boolean* indikátory. Tyto indikátory mají defaultní barvu zelenou, a pokud jsou sepnuty, začnou svítit červeně.

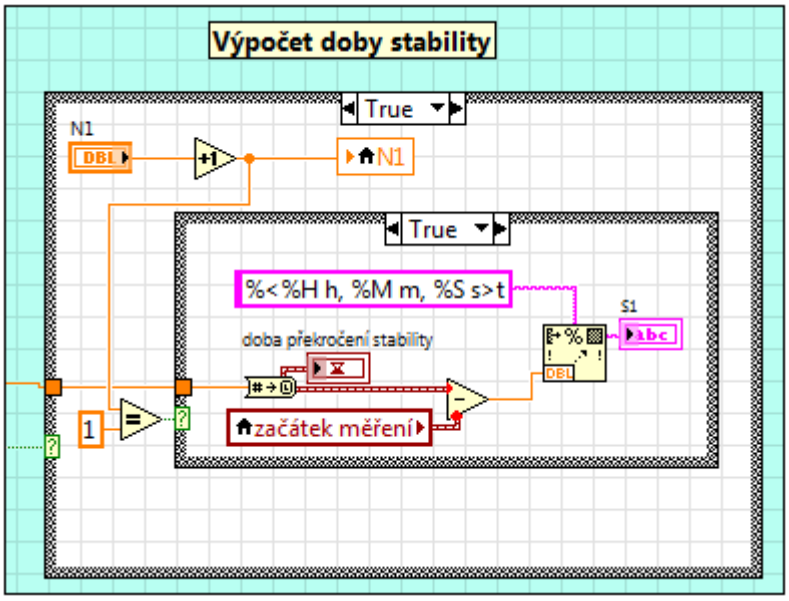


Obrázek 53: Bloky pro indikování překročení aerobní stability

11.4.11 Popis bloků pro výpočet doby stability

Jedním ze základních požadavků na program bylo, aby dokázal vypočítat dobu, po kterou zůstává vzorek měřeného krmiva stabilní. Bloky umožňující výpočet času stability jsou vidět na obrázku 54. Základem jsou dvě *case* struktury. Do vnější *case* struktury vstupuje aktuální systémový čas a datový typ *boolean*. Tento je stejný jako ten, který ovládá *case*

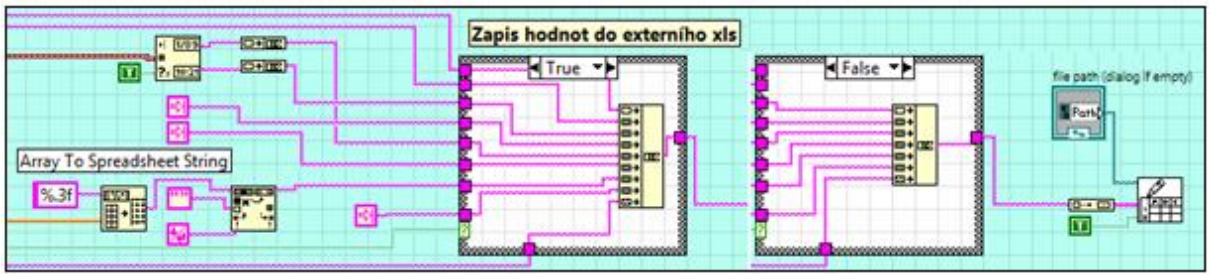
strukturu popsanou v předchozí kapitole a je veden od bloku *Or*, který můžeme vidět na obrázku 53. Pokud je pro konkrétní čidlo indikováno překročení stability LED indikátorem na čelním panelu je vypočtena i doba stability vzorku měřeného tímto čidlem. Vnitřní *case* je ovládán podmínkou, která zaručuje, že vypočtený čas nebude přepisován a doba stability, tak vlastně prodlužována. Jakmile je stabilita překročena je prvek *Numeric kontrol* nastaven na hodnotu 1 a ta je dále porovnána pomocí bloku *Equal*. Při tomto překročení je druhý *case* nastavena na hodnotu *true* a je vypočten čas. Při dalším překročení, ale už hodnota prvku *numeric* není rovna jedné, a proto už nedojde k přepisování vypočteného času stability. Ve druhé *case* struktuře je potom od doby překročení stability odečten čas začátku měření. Výsledná hodnota vstupuje do bloku *Format into string*, který tuto číselnou hodnotu v sekundách dokáže převést na čas ve formátu hodin, minut a sekund. Tato hodnota je spolu se systémovým časem překročení stability zobrazena na čelním panelu pod záložkou stabilita. Tímto způsobem je čas stability počítán pro čidla 1 až 15.



Obrázek 54: Bloky pro výpočet doby stability

11.4.12 Popis ukládání naměřených dat do externí tabulky

Důležitým bodem zadání je možnost průběžného ukládání naměřených dat do souboru typu xls umístěného na disku počítače. Tato část programu ochrání data pro případ pádu systému. Popis bloků, které toto umožňují, je rozdělen do několika obrázků. První část je na obrázku 55. V pravé části je blok *Write to spreadsheet file*, jež přímo ukládá hodnoty v textovém řetězci do externí tabulky. Do tohoto bloku vede ovládací prvek zobrazený na čelním panelu, pomocí kterého můžeme zvolit umístění a název externího souboru. Dále do něj vede *boolean* konstanta, která blok nastavuje, tak aby se jednotlivá měření dále připisovala do stejného výstupního souboru. Hodnoty vstupující do tohoto bloku je nejprve potřeba vhodně naformátovat a převést do textového řetězce.

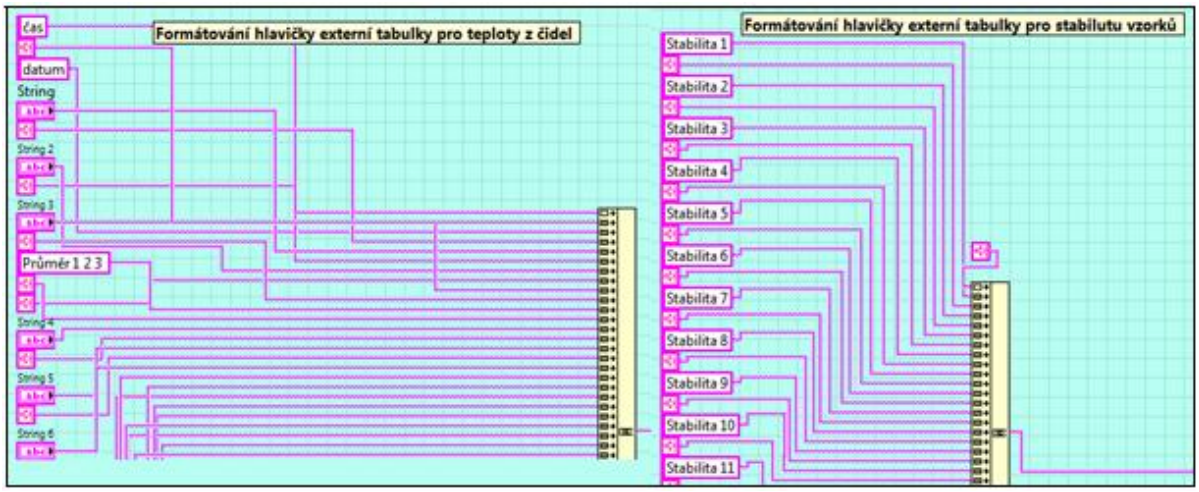


Obrázek 55: Bloky pro zápis do externího xls

Do *case* struktury vstupují textové řetězce formující hlavičku externí tabulky popsané níže. Dále sem vstupují samotné hodnoty. Nejprve datum a čas, které jsou na textový řetězec převedeny z časové značky pomocí bloku *Get Date/Time String*. Potom hodnoty jednotlivých teplot převedeny na textový řetězec, ale pomocí bloku *Array To Spreadsheet String*. U tohoto řetězce se vyskytoval prázdný řádek, který byl odstraněn pomocí bloku *Search And Replace String*. Poslední vstupují do *case* struktury hodnoty doby stability jednotlivých čidel, jejichž naformátování je zobrazeno na obrázku níže.

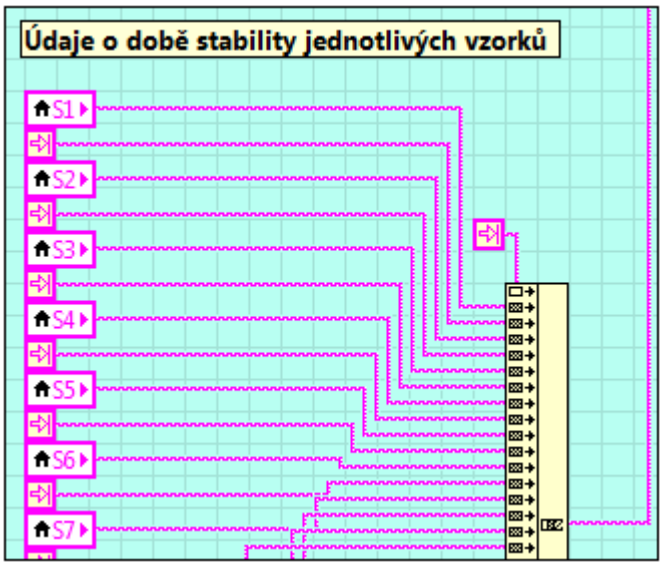
Case struktura má za úkol pro první měření vložit do externího souboru námi naformátovanou hlavičku. V dalších krocích měření už tuto hlavičku nevpisuje, ale ukládá pouze řádky s naměřenými hodnotami. Pro první měření je pomocí podmínky nastavena hodnota na *true* a program vepíše hlavičku. Pro všechna ostatní měření je hodnota nastavena na *false*, jak je zobrazeno na obrázku 55, a do tabulky se vepisují už jen naměřená data.

K formátování hlavičky externí tabulky slouží bloky zobrazené na obrázku 56. Textové konstanty a proměnné jsou odděleny tabulátorovou konstantou, která je rozděluje do jednotlivých sloupců a v určeném pořadí vstupují do bloku *Concatenate string*, který je spojuje do jednoho řetězce. Tyto řetězce následně vstupují do bloků zobrazených na obrázku 55.



Obrázek 56: Formátování hlavičky externí tabulky

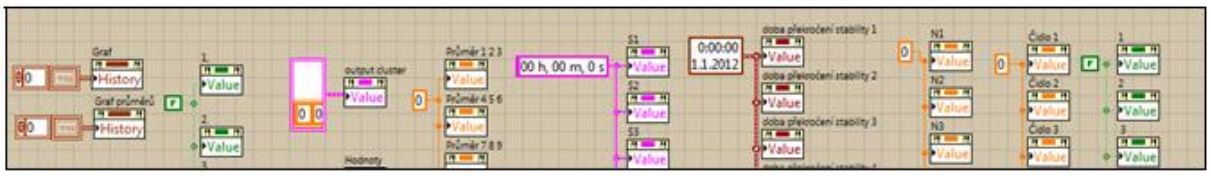
Obrázek 57 níže zobrazuje ukládání hodnot o naměřených dobách stability do textového řetězce. Lokální proměnné s údaji o době stability jsou odděleny tabulátorovou konstantou a v daném pořadí vstupují do bloku, který je převede na textový řetězec. Ten potom vstupuje do dalších bloků a je ukládán do externí tabulky.



Obrázek 57: Formátování údajů o době stability vzorků

11.4.13 Popis bloků pro nulování hodnot indikátorů

Aby se jednotlivá naměřené hodnoty nezobrazovaly při opětovném spuštění programu, je potřeba tyto hodnoty po spuštění nulovat. To provádí bloky zobrazené na obrázku níže. V programu se vybere lokální proměnná pro počáteční hodnotu a přiřadí se k ní nulová konstanta. Takto se nulování provede jak pro numerické indikátory, tak i pro grafy, textová pole, časové značky, ale i LED indikátory.



Obrázek 58: Bloky pro nulování hodnot

11.5 Vzhled externí tabulky

Program *Temper.vi* ukládá naměřené hodnoty do externí tabulky zobrazené níže. Pomocí bloku popsaných v kapitole 11.4.12 lze libovolně formátovat hlavičku sloupečků pro názvy vzorků. Hlavičky čas, datum, hlavičku průměrů a AMBIENT není možné formátovat, protože je zadána pomocí textových konstant.

Tabulka 5: Naměřené hodnoty systémem pro měření aerobní stability

čas	datum	vzorek 1	vzorek 2	vzorek 3	Průměr 1 2 3	…..	vzorek 15	průměr 13 14 15	AMBIENT
10:36:55	13.5.2012	22,00	22,00	22,00	22,00	…..	22,00	22,00	22,00
10:37:17	13.5.2012	21,13	18,94	18,13	19,40	…..	22,19	19,67	18,06
10:37:39	13.5.2012	24,81	18,81	19,19	20,94	…..	26,88	21,52	18,75
10:38:01	13.5.2012	28,00	19,00	19,06	22,02	…..	29,13	22,31	18,94
10:38:23	13.5.2012	28,94	19,19	19,88	22,67	…..	28,88	22,38	19,13
10:38:46	13.5.2012	27,19	19,88	20,13	22,40	…..	27,13	22,10	18,75
10:39:08	13.5.2012	26,88	20,00	19,81	22,23	…..	26,75	21,79	18,81

Další část výstupní tabulky zobrazuje údaje o době aerobní stability čidel 1 až 15. V našem příkladě v tabulce dole můžeme vidět, že stabilitu překročilo jen čidlo číslo 2, po 3 minutách a 47 sekundách. V této tabulce se hodnoty času stability přepisují neustále stejně, a proto se po skončení měření výsledné údaje vyčtou až v nejnižším řádku tabulky.

Tabulka 6: Tabulka zobrazující dobu stability

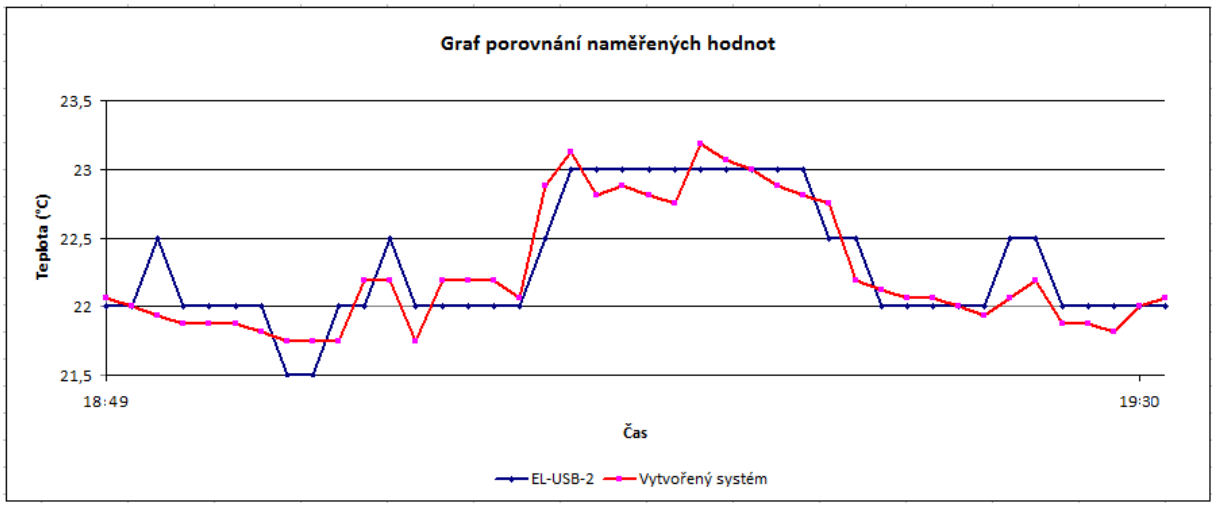
Stabilita 1	Stabilita 2	Stabilita 3	Stabilita 4
00 h, 00 m, 0 s	00 h, 00 m, 0 s	00 h, 00 m, 0 s	00 h, 00 m, 0 s
00 h, 00 m, 0 s	00 h, 00 m, 0 s	00 h, 00 m, 0 s	00 h, 00 m, 0 s
00 h, 00 m, 0 s	00 h, 00 m, 0 s	00 h, 00 m, 0 s	00 h, 00 m, 0 s
00 h, 00 m, 0 s	00 h, 03 m, 47 s	00 h, 00 m, 0 s	00 h, 00 m, 0 s
00 h, 00 m, 0 s	00 h, 03 m, 47 s	00 h, 00 m, 0 s	00 h, 00 m, 0 s
00 h, 00 m, 0 s	00 h, 03 m, 47 s	00 h, 00 m, 0 s	00 h, 00 m, 0 s

12. Testování naměřených hodnot

Kontrola správnosti naměřených hodnot byla provedena pomocí komerčního teplotního dataloggeru zobrazeného na obrázku 59. Jedná se o produkt EL-USB-2, který zaznamenává teplotu v nastavitelných intervalech a ukládá zaznamenané hodnoty do textového souboru. Nevýhodou tohoto systému je, že zaznamenává teploty s přesností pouze na 0,5°C.

Obrázek 59: USB datalogger EL-USB-2 [16]

Bylo provedeno měření, které trvalo 40 minut. Komerční dataloger byl umístěn hned vedle čidel našeho systému a byl spuštěn záznam. Naměřené hodnoty jsou porovnané v grafu na obrázku níže. Můžeme vidět, že hodnoty naměřené vytvořeným systémem odpovídají hodnotám naměřeným komerčním zařízením.



Obrázek 60: Graf porovnávající naměřené hodnoty

13. Závěr

Cílem diplomové práce bylo vytvořit systém pro měření aerobní stability, který by vyhovoval požadavkům firmy Nutrivet s.r.o. Úkolem bylo seznámit se s teorií výroby fermentovaných krmiv a aspekty ovlivňující především aerobní stabilitu těchto krmiv. Výrobě krmiv je věnováno několik stran textu. Podrobně jsme se seznámili především s jednotlivými procesy výroby a konzervace, ale také s důsledky nesprávné konzervace a procesů ovlivňující sekundární fermentaci a tím i aerobní stabilitu krmiv.

Dále bylo úkolem navrhnout systém, který by umožňoval měřit právě dobu stability těchto krmiv. Pro realizaci bylo využito teplotních čidel DS18S20, která splňují požadavky pro toto měření. Jmenovitě mají ideální rozsah teplot pro měření, dostatečnou přesnost a také rozměry umožňující jejich zapouzdření do vpichových sond. Tento tvar je ideální protože krmiva se při analýze umísťují do malých kontejnerů válcovitého tvaru. Krmivo je v těchto kontejnerech natlačeno a vpichové sondy umožňují jednoduché zavedení čidla do středu kontejneru.

Teplotní čidla jsou propojena prostřednictvím 1-wire sítě. Aby bylo možné číst teploty ze všech čidel, byl vyroben přípravek, který umožňuje připojit čidla do počítače přes sériový port. Tento přípravek vychází z obvodu převodníku DS9097E vyrobeným firmou Dallas Semiconductor. Přípravek nevyžaduje napájení, protože 1-wire sběrnice je napájena přímo přes sériový port. Veškeré součástky byly napájeny na universální pájivé pole, včetně šestnácti konektorů pro vpichové sondy. Počet sond byl zadán firmou Nutrivet,s.r.o. Hotový přípravek byl uložen do plastového boxu, zabraňující vniknutí vlhkosti a nečistot.

Dalším bodem zadání bylo vytvořit program ve zvoleném programovacím prostředí. Bylo vybráno prostředí LabVIEW, které je vhodné zejména pro jeho možnost pracovat s daty ze sériového portu, názornou a přehlednou vizualizaci dat a ovládacích prvků a možnost zálohování dat do externího xls souboru. Program umožňuje zpracování dat z teplotních čidel a jejich přehlednou reprezentaci na čelním panelu programu. Uživatel může zadat interval jednotlivých měření. Čelní panel zobrazuje hodnoty patnácti měřících čidel pro vzorky a hodnoty naměřené ambientním čidlem, které měří teplotu okolního prostředí. Dále je zde zobrazen graf zobrazující vývoj teploty měřících čidel v čase včetně prahu indikující překročení hranice teplotní stability a také graf zobrazující průměr vypočítaný z hodnot jednotlivých trojic čidel. Čelní panel programu indikuje pomocí LED prvků překročení stability a zobrazuje hodnoty doby stability naměřené jednotlivými měřícími čidly.

Program také umožňuje naměřené hodnoty průběžně ukládat do externí tabulky. Hlavička této tabulky může být formátována a tak se stává uživatelsky mnohem přehlednější a použitelnější.

Správnost dat byla nakonec ověřena pomocí komerčně vyráběného teplotního dataloggeru a porovnána s daty naměřenými vytvořeným systémem.

Možnost vylepšení systému vidím především v rozšíření počtu teplotních čidel, tak aby se jedním měřením vyhodnotili více vzorků a celý systém by tak byl ekonomičtější. Dále tu je možnost systém upravit a použít jej v monitorování vývoje teploty ve velkých systémech jako jsou například strojovny, laboratoře a serverovny. Možnost by byla také v indikaci překročení alarmů a následné regulaci teploty takto kontrolovaných systémů.

14. Použitá literatura

[1] *Co je to siláž? Jak připravit siláž? Které plodiny jsou vhodné pro silážování? | Zemědělské potřeby M+S s.r.o.* [online]. 2011 [cit. 2011-12-02]. Co je to siláž. Dostupné z WWW: <http://www.akaska.cz/sdruzeni-ms/co-je-silaz.php>.

[2] LOUČKA, Radko. Ekonomika výživy skotu . *Náš chov*. 2011, 6. [cit. 2011-11-22].

[3] Příprava pokusných mikrosiláží. In VYSKOČIL, Ivo. *Příprava pokusných mikrosiláží* [online]. VZ 1.07. [s.l.] : [s.n.], 2010 [cit. 2011-11-24]. Dostupné z WWW: <http://web2.mendelu.cz/af_291_projekty/files/6/6-mikrosilazovani.pdf>.

[4] RADA, Vojtěch. Siláž a zdraví zvířat. In *Siláž a zdraví zvířat* [online]. Praha : [s.n.], 2009 [cit. 2011-12-05]. Dostupné z WWW: <http://www.vuzv.cz/sites/SilazRada.pdf>.

[5] KUNG, L. Aerobic Stability of Silage, The 2010 California Alfalfa and Forage Symposium, Visalia, California.

[6] MAXIM DS18S20 High-Precision 1-Wire Digital Thermometer datasheet [online], Rev 8/10 [cit. 3. Prosince 2011]. Dostupné z <http://datasheets.maxim-ic.com/en/DS18S20.pdf>.

[7] Siláž. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, , last modified on 29.7.2011 [cit. 2011-12-05]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Siláž>.

[8] HONIG, H. *Evaluation of aerobic stability.* Braunschweig, 1990 [cit. 2011-12-05]

[9] Overview of 1-Wire® Technology and Its Use [online], [cit. 3. Prosince 2011]. Dostupné z < http://pdfserv.maxim-ic.com/en/an/AN1796.pdf>.

[10] MicroLAN Design Guide [online], [cit. 5. Prosince 2011]. Dostupné z < http://www.datsi.fi.upm.es/docencia/Micro_C/dallas/tb1.pdf>.

[11] JAMBOR, Václav. Proč se glycidové siláže zahřívají. In: 2004. Pohořelice, 2004. [cit. 2011-11-22].

[12] Agro24: Silážní vaky. [online]. [cit. 2011-11-23]. Dostupné z: http://www.agro24.eu/agro24-eu/eshop/2-1-Silazni-vaky/0/5/87-Silazni-vak-2-7m-x60m-x-240-micronu

[13] ZDBudiskovice: Budovy. [online]. [cit. 2011-11-23]. Dostupné z: http://zdbudiskovice.nolimit.cz/budovy-2

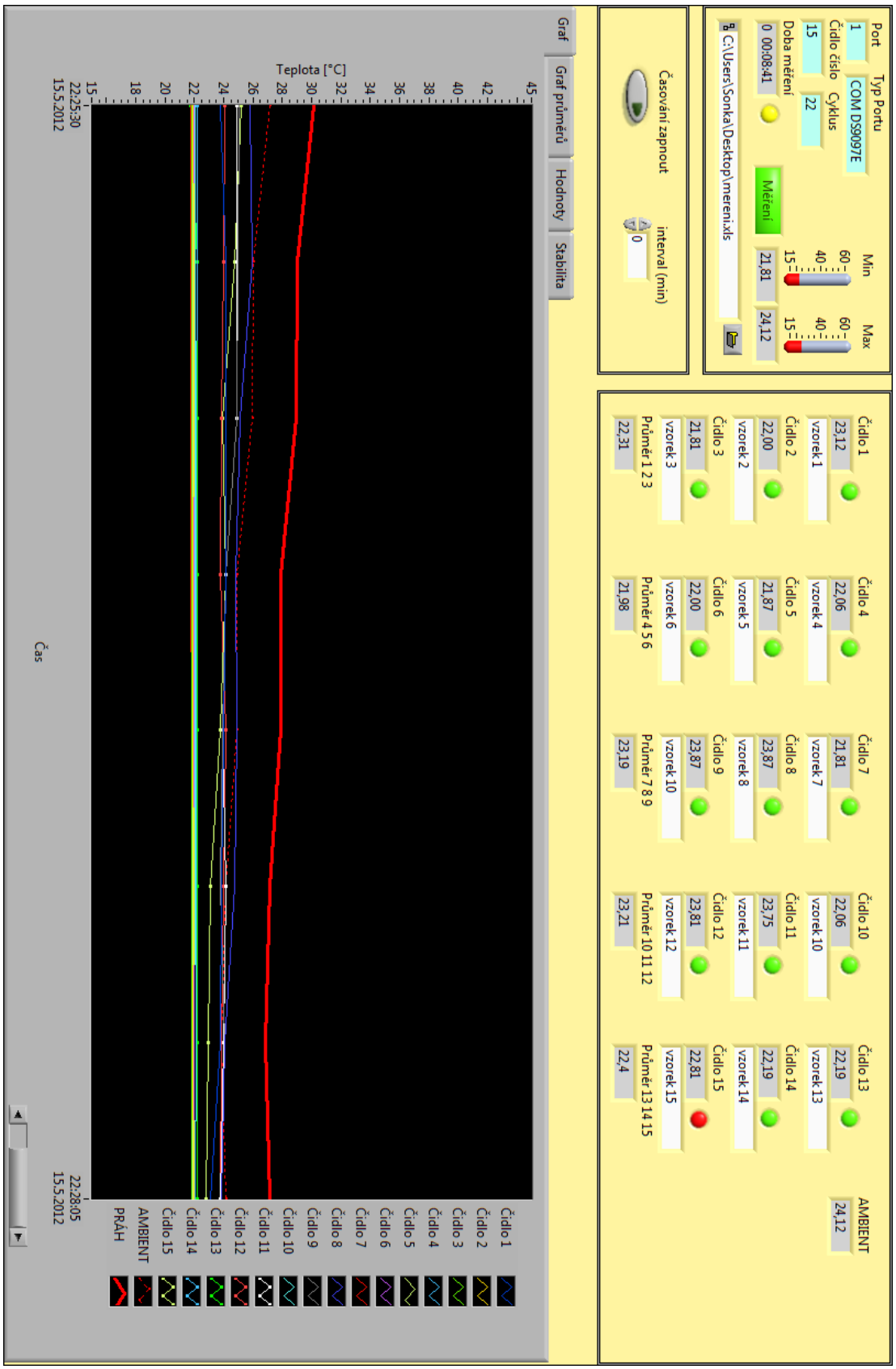
[14] POTYŠ, J. Měření a analýza neharmonických signálů v programu LabVIEW, VUT Brno

[15] TMEX API List by Group. [online]. [cit. 2012-05-10]. Dostupné z: http://files.dalsemi.com/auto_id/softdev/owdocs/Docs/TMEX/tmex3vlg.html

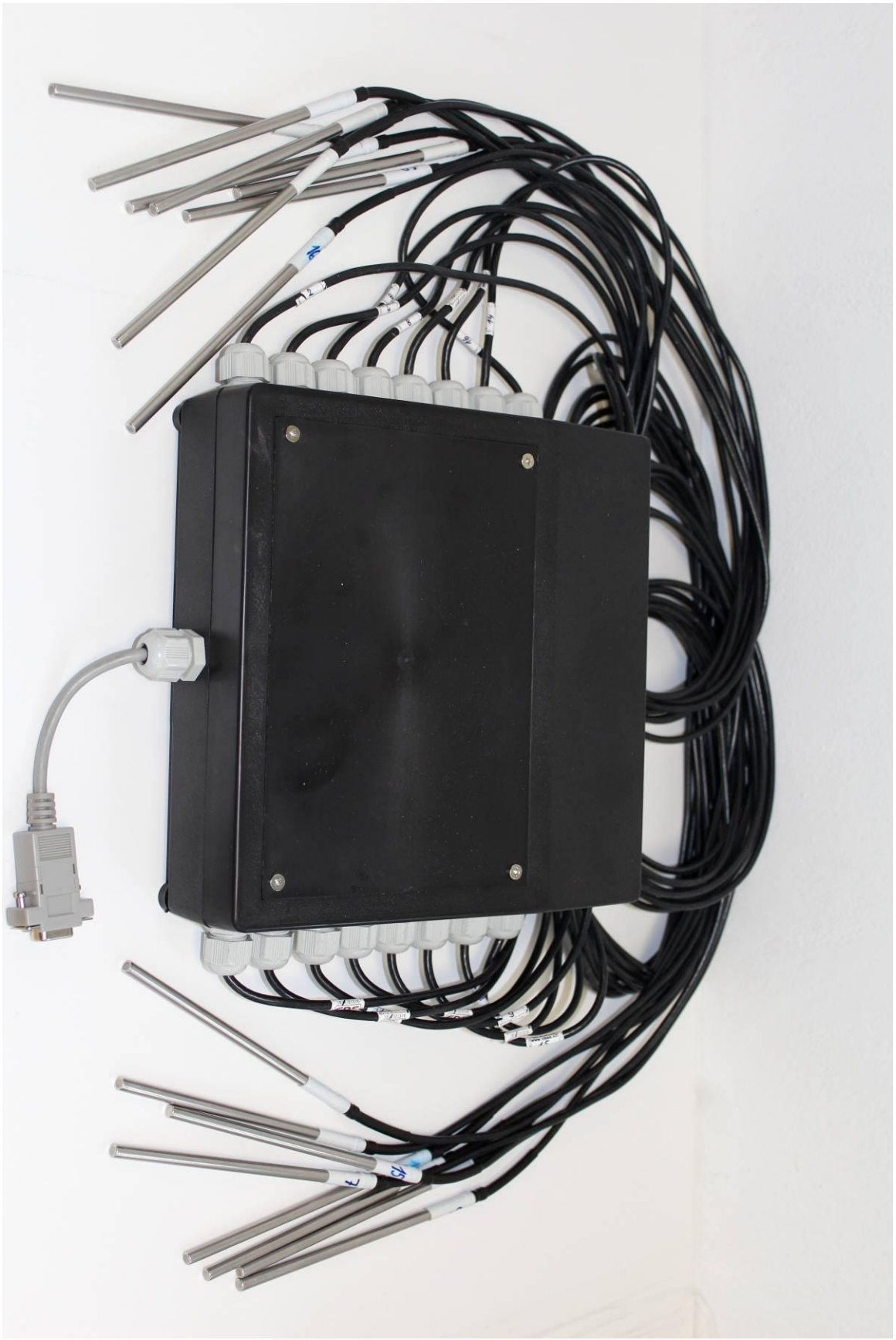
[16] DATALOGGER, RH TEMP - EL-USB-2 - LASCAR. [online]. [cit. 2012-05-13]. Dostupné z: http://cpc.farnell.com/1/1/45779-datalogger-rh-temp-el-usb-2-lascar.html

15. Přílohy

Příloha 1: Program *Temper.vi* – kompletní čelní panel



Příloha 2: Fotografie přípravku spolu se všemi čidly



Příloha 3: Detailní fotografie přípravku